Universidad de las Ciencias Informáticas

Facultad 1

Título: Componentes de Publicaciones y Premios del Sistema de Investigaciones

Trabajo de Diploma para optar por el Título de

Ingeniero en Ciencias Informáticas.

Autores:

Angel Antonio Sánchez López-Castro.

Marycarmen Díaz Labrador

Tutores:

Ing. Ivis Cañizares Rivera.

Ing. Alex Rodríguez Serrano.

Ing. Marielí Barrera Fabregat.

La Habana, 2015

¨Año 57 de la Revolución¨

Declaración de autoría

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmamos las presentes a los ____ días del mes de _____ del año 2015.

Autores:

_____ _____

Marycarmen Díaz Labrador Angel Antonio Sánchez López-Castro

Tutores:

_____ _____

Ing. Ivis Cañizares Rivera Ing. Alex Rodríguez Serrano

_____

Ing. Marielí Barrera Fabregat

Marycarmen Díaz Labrador

Cuando bebas agua, recuerda la fuente.

Agradecerle a Mamá su cariño y dedicación infinito. Papá por dar todo de sí, incluso cuando no podía. Mi abuela Leonor por ser mi súper abuela como yo le digo. Mi abuelo Juan que siempre me exigía mucho en español. Mi abuelo Cosme que siempre está pendiente de mi hermano y de mí. Mi hermano Luis Angel que aunque nos peleamos y nos molestamos, el amor que nos une está siempre presente hasta en los tiempos difíciles. Mi novio Antonio que en los 7 años, 6 meses y 18 días que llevamos como amigos, compañeros y amantes; he aprendido tanto de él y de mi misma, nos ha cambiado a ambos. Gracias por ALL y TE AMO. Mis suegros Mari y Toni, gracias a ambos por quererme como a una hija. Mi cuñado y Yaque por alimentarnos y cuidarnos en los tiempos de tanta hambre en la escuela. Mi tío Dago te quiero, me he sentido siempre muy querida por ti. El resto de mi familia, el apoyo de ellos siempre ha sido es esencial.

Mi compañero de tesis siempre muy preocupado, es lo que más me gustaba de él y nos entendimos muy bien durante esta etapa, gracias Angel, hemos sido un equipo, nos caímos y nos levantábamos pero siempre para adelante. Mi amiga de toda la vida Evelyn son tantos años ya y siempre estamos ahí. Mis tutores Ivis, Marielí y Alex y especialmente a Alex que ha estado presente en todo momento, gracias por tu paciencia y dedicación y por todo lo que me has enseñado. Mi gente del proyecto que aunque no eran mis tutores siempre estuvieron presentes cuando los necesitamos. En especial a Mercedes, Yasmany, Yanio y Alexander. Mis compañeros de aula, mis amistades. Han sido cinco años maravillosos donde muchas personas han sido parte de mi vida y mi crecimiento como persona y profesional. Gracias a todos.

Angel Antonio Sánchez López-Castro

Agradecerle a mi mamá por haberme traído al mundo y por todo el apoyo que me ha brindado en toda mi vida, especialmente en esta etapa que ha sido bien difícil. A Robe que lo ha hecho todo por nosotros y para que todo nos haya ido bien, quiero que sepas que eres mi papá también. A papi por estar siempre presente a pesar de la distancia o las dificultades, disculpa por haberte hecho correr tanto. A mi tío Migue que desde que era un niño siempre ha sido un motor impulsor en mi vida y me ha ayudado mucho en todo. A mi hermana, Malanga te quiero mucho y eres mi vida, gracias por haber estado ahí siempre. A Katy que es mi mujer, mi amiga, mi compañera, que me ha apoyado en todo y me dio un regalo precioso: Vanessa, las amo.

A mi compañera de tesis, Mary muchas gracias por todo y por ser tan paciente conmigo, siempre nos llevamos muy bien y supimos salir adelante. A mis tutores Ivis, Marielí y especialmente a Alex, gracias por toda tu ayuda y por la paciencia que tuviste con nosotros. A mis compañeros de cuarto Jorge F, Jorge Angel, Manolo y Osmel por haber estado siempre en mis momentos buenos y malos, los quiero y son como una familia para mí. A mis hermanos Benito, Alexo, Sánchez, David, Pedrito, Eduardo, Estrada, Robe, el ruso, que aunque me dejaron un año por detrás menos Sánchez nunca han fallado y siempre han sido transparentes. A mis compañeros de grupo en general, a Orli, Angelito, el guille, al tierno y especialmente al Equipo 1: Abelito, Ively, Osmel y Carlitos, gracias por su tiempo y su paciencia conmigo, siempre los tendré presente.

Resumen

En el área de investigaciones de la Universidad de las Ciencias Informáticas se organizan las actividades científicas, donde se gestionan los premios y certifican las publicaciones. En la realización de ambos procesos se identificó que la información generada se almacena utilizando herramientas ofimáticas, dificultando su procesamiento. Además el intercambio de información se realiza a través de correo electrónico, provocando que existan duplicados o pérdida de la información, por lo que se hace compleja su manipulación y almacenamiento. Estas deficiencias provocan demora en la entrega de los premios y la certificación, por lo que se plantea como objetivo de la investigación desarrollar los componentes Publicaciones y Premios del Sistema de Investigaciones con tecnologías libres, que permitan agilizar los procesos de premios curriculares y certificación de publicaciones. Para darle cumplimiento al objetivo propuesto, se realizó un estudio a nivel nacional e internacional con el fin de conocer cómo se realiza la gestión de los premios y la certificación de publicaciones. Luego, se identificaron los requisitos funcionales y no funcionales y se efectuó el diseño e implementación, así como la ejecución de pruebas a los componentes. La solución desarrollada permite agilizar los procesos de premios y certificación de publicaciones gestionados por la Dirección de Investigaciones.

Palabras clave: certificación de publicaciones, componente, Dirección de Investigaciones, premio.

Índice general

Introducción

Las tecnologías de la información y las comunicaciones (TIC) juegan un papel importante en el desarrollo de la sociedad, han contribuido a acceder de forma dinámica e inmediata a la información, así como establecer nuevas estructuras de comunicación. Actualmente se han convertido en un eslabón fundamental para la gestión de la información, que exigen la innovación, renovación y el perfeccionamiento de herramientas, procesos y productos. Cuba no se ha quedado al margen de este desarrollo tecnológico. El país realiza grandes esfuerzos por lograr la informatización de varios sectores de la sociedad cubana, dando pasos significativos para alcanzar este propósito.

Uno de los ejemplos más representativos es la creación de la Universidad de las Ciencias Informáticas (UCI), para liderar el desarrollo de software en el país. La UCI fue creada en el año 2002 con la misión de producir aplicaciones y servicios informáticos, así como, servir de soporte en el proceso de informatización del país (1). Las investigaciones en la universidad potencian los resultados en la producción de software y la formación (2). El elevado trabajo investigativo que se genera provoca que las categorías científicas y docentes requieran expedientes con documentación que acredite los indicadores de producción científica: participación en eventos científicos, proyectos de investigación y desarrollo (I+D) y premios científicos.

A pesar de que la universidad cuenta con los elementos tecnológicos necesarios para informatizar los procesos, no se aprovechan del todo para la gestión de las actividades científicas que organiza la Dirección de Investigaciones (DI). Actualmente en la universidad se desarrolla el Sistema de Gestión Universitaria como solución integral para la gestión de los procesos sustantivos de la universidad, este se encuentra integrado por los sistemas: Pregrado, Postgrado, Cooperación, Residencia, Teleformación, Biblioteca, Desarrollo, Tecnologías, Investigaciones, Extensión, Organizaciones y Egreso. Sin embargo el sistema que tributa al área de Investigaciones no gestiona las actividades de investigación científica organizadas y desarrolladas por la DI.

La DI dirige metodológicamente las investigaciones en la UCI y ejecuta acciones concretas relacionadas con la Ciencia, Tecnología e Innovación (CTI). Se encarga de las diferentes líneas de investigación, gestiona las actividades científicas, establece alianzas con diferentes instituciones nacionales e internacionales. Además coordina, garantiza la logística y apoya con capital humano en la realización de eventos organizados por la universidad o por otra entidad. También promueve eventos científicos, certifica las publicaciones y gestiona los premios curriculares tanto para estudiantes como trabajadores.

Las investigaciones científicas se realizan para solucionar problemáticas, innovar o sencillamente para aumentar el intelecto. Estas en muchas ocasiones llegan a ser publicadas por diversas entidades. Para tener un control de las publicaciones realizadas por la comunidad universitaria, se realiza un proceso de certificación de publicaciones, actualmente es una sola persona la encargada de recibir por correo electrónico las solicitudes. Por el gran cúmulo que estas representan, el proceso se hace engorroso provocando demora en la certificación por lo que la solicitud es enviada en reiteradas ocasiones, implicando que existan duplicados. De ser verificada la existencia de la publicación se confecciona un certificado en la biblioteca, este debe ser firmado en la DI por el director en el momento que disponga, luego de ser impreso en el departamento de impresión. Esto implica que el certificado transite por varios departamentos que se encuentran en diferentes áreas de la universidad, haciendo que el proceso se extienda y retrase la entrega. La impresión del certificado se realiza siempre, pero en varias ocasiones no es recogido por el investigador y después de un período prolongado es desechado, de esta forma se invierte tiempo y recursos innecesariamente para la emisión del certificado.

La DI para la entrega de premios se rige por la Resolución No. 1/2009 del Ministerio de Educación Superior (MES); estos reconocimientos han contribuido a la elevación de la calidad y productividad de la labor científica de investigadores. Los premios gestionados, se dividen según su alcance: nacionales y UCI. Entre los nacionales se pueden mencionar los premios CITMA, otorgados por el Ministerio de Ciencia, Tecnología y Medio Ambiente, entre los que se encuentran: el premio Nacional Anual para Jóvenes Investigadores y el premio Nacional Anual para Estudiantes Investigadores. Este último está dirigido a reconocer la labor de los jóvenes estudiantes de alto aprovechamiento docente que se destacan en el trabajo científico-técnico en los Centros de Educación Superior.

Los premios poseen varias ediciones y se realizan con una frecuencia determinada (trimestral, semestral o anual). A partir de la convocatoria se conforma un expediente para cada propuesta, escogidas según la labor destacada en las actividades de CTI. En ocasiones la confección del expediente no se realiza de forma correcta, ya que no existe un formato y estructura definida para su confección, provocando en determinados casos que la misma información pueda tener varias versiones y formatos, en dependencia de la persona que la gestiona. Estos son almacenados en las distintas áreas y en la DI, lo que trae consigo que la información se encuentre duplicada en ambos niveles. Para elegir los premiados no existe un mecanismo que permita diferenciar cuantitativamente los parámetros de evaluación para cada persona, propiciando que la selección se torne engorrosa y propensa a errores. La DI convoca un encuentro entre los miembros de la

comisión, que en muchas ocasiones se vuelve a planificar debido a que los integrantes son de distintos departamentos y tienen otras responsabilidades, provocando que se extienda el proceso de evaluación.

En los procesos de premios y certificación de publicaciones mencionados, se detectaron de forma general las siguientes deficiencias:

- La información que se genera se almacena a través de las aplicaciones ofimáticas y debido al gran cúmulo que esta representa, dificulta el procesamiento de la misma por el personal encargado de su análisis.
- No existe un registro para las consultas de publicaciones y premios de los estudiantes y trabajadores de la universidad.
- El intercambio de información, dígase solicitudes, notificaciones y resultados se realizan a través del correo electrónico, lo que hace compleja la manipulación y almacenamiento, afectándose así la agilidad y calidad del proceso, provocando que existan duplicados o pérdida de la información, así como dificultades en la gestión manual de grandes volúmenes de datos.
- Actualmente no se encuentran contemplados los procesos en el Sistema de Gestión Universitaria.

Teniendo en cuenta los planteamientos anteriores, se deriva el siguiente **problema de investigación:**

¿Cómo agilizar los procesos de premios curriculares y certificación de publicaciones para la gestión de las actividades de investigación científica organizadas por la Dirección de Investigaciones en la Universidad de las Ciencias Informáticas?

Constituye el **objeto de estudio** los procesos de premios y publicaciones, enmarcado en el **campo de acción** los procesos de premios curriculares y certificación de publicaciones en la Universidad de las Ciencias Informáticas.

Para resolver el problema planteado se propone como **objetivo general**: desarrollar los componentes Publicaciones y Premios del Sistema de Investigaciones con tecnologías libres, que permitan agilizar los procesos de premios curriculares y certificación de publicaciones organizados por la Dirección de Investigaciones en la Universidad de las Ciencias Informáticas.

De acuerdo con la propuesta anterior se trazan los siguientes **objetivos específicos**:

- Caracterizar los principales elementos teóricos de los procesos de premios curriculares y certificación de publicaciones.
- Diseñar los componentes Publicaciones y Premios del Sistema de Investigaciones.
- Implementar los componentes Publicaciones y Premios del Sistema de Investigaciones.
- Evaluar las funcionalidades de los componentes Publicaciones y Premios mediante pruebas de *software*.

Idea a defender:

El desarrollo de los componentes Publicaciones y Premios del Sistema de Investigaciones, agilizará la gestión de las actividades de investigación científica organizadas por la Dirección de Investigaciones en la Universidad de la Ciencias Informáticas.

Para el cumplimiento del objetivo de la investigación se han propuesto las siguientes **tareas de la investigación:**

- Análisis de los diferentes conceptos relacionados con la investigación.
- Caracterización de los negocios de premios curriculares y certificación de publicaciones.
- Caracterización de las herramientas a utilizar en el entorno de desarrollo.
- Análisis de técnicas de ingeniería de requisitos.
- Identificación de los requisitos funcionales y no funcionales.
- Definición de los patrones de diseño para la construcción de los componentes.
- Modelado de la base de datos.
- Implementación de los requisitos especificados para el desarrollo del sistema.
- Diseño de las pruebas a realizarle a los componentes desarrollados.
- Realización de las pruebas del sistema.

Para la realización de la presente investigación, se hace necesaria la utilización de los siguientes métodos científicos:

Métodos Teóricos:

- Histórico-Lógico: permite estudiar de forma razonada la trayectoria histórica real de los fenómenos, su evolución y desarrollo (3). Se realiza un estudio de los sistemas de gestión de información relacionados con publicaciones y premios, así como las tecnologías que existen actualmente.
- Analítico-Sintético: son dos procesos inherentes al pensamiento, operaciones lógicas importantes; que permiten como métodos teóricos, buscar la esencia de los fenómenos, rasgos que los caracterizan y distinguen. Su objetivo en una investigación es analizar las teorías, documentos, permitiendo la extracción de los elementos más importantes que se relacionan con el objeto de estudio (3). Se utiliza para el análisis de teorías y documentos, se extraen los elementos más importantes de cada uno de los aspectos esenciales de las herramientas y la literatura seleccionada para generar los certificados de publicaciones para los componentes Publicaciones y Premios del área de Investigaciones en el Sistema de Gestión Universitaria de la Universidad de las Ciencias Informáticas.

Método Empírico:

Entrevista: es una conversación planificada para obtener información. Su uso constituye un medio para el conocimiento cualitativo de los fenómenos o sobre características personales del entrevistado y puede influir en determinados aspectos de la conducta humana, por lo que es importante una buena comunicación (4). Permite realizar entrevistas con el cliente para obtener la mayor cantidad de información posible, entender todo el proceso de negocio, comprender la estructura y funcionamiento de los procesos actuales, así como los problemas existentes, permitiendo definir específicamente el problema a resolver y establecer el objeto de estudio. Para consultar la entrevista dirigirse al Anexo 1.

Posible resultado:

- ✓ Documentación ingenieril asociada al negocio, requisitos, diseño e implementación.
- ✓ Componentes de Publicaciones y Premios que tributen a la gestión de estos procesos en la Dirección de Investigaciones de la UCI.

Estructura del documento

El presente trabajo está estructurado en tres capítulos que abarcan todo el proceso para el desarrollo de la solución informática.

Capítulo 1: Fundamentación teórica: en este capítulo se presentan los elementos teóricos que se utilizarán como base en la investigación. Se hace una descripción de los conceptos fundamentales asociados al dominio del problema. Se describen los procesos de premios y certificación de publicaciones en la UCI y en el mundo. Se define el entorno tecnológico en el desarrollo para la propuesta de solución.

Capítulo 2: Descripción de la propuesta de solución: en este capítulo se realiza una breve descripción del flujo de negocio y la propuesta de solución, sus requisitos funcionales y no funcionales. Se describe el estilo arquitectónico del sistema, haciendo énfasis en los patrones de diseño y de base de datos utilizados. Se presenta el modelo de datos y la distribución física o diagrama de despliegue de dicha propuesta.

Capítulo 3: Validación de la propuesta de solución: en este capítulo se describe la implementación de la propuesta de solución teniendo en cuenta los estándares de codificación empleados. Se presentan las pruebas realizadas que servirán para validar la propuesta de solución como las pruebas unitarias, de integración, del sistema y de aceptación para satisfacer todas las necesidades del cliente.

Capítulo 1: Fundamentación teórica

En el presente capítulo se definen los principales conceptos asociados al dominio de la problemática existente y se realiza una descripción del objeto de estudio de la investigación. Se brinda una breve panorámica sobre algunos de los sistemas informáticos vinculados al objeto de estudio. Se caracteriza el proceso de desarrollo, las técnicas de programación y las principales tecnologías utilizadas para el desarrollo de la solución informática, definidas por el Grupo de Arquitectura de la Dirección de Informatización.

1.1 Conceptos asociados

1.1.1 Investigación científica

El estudio de la investigación científica no es exclusivamente de este siglo, se remonta a los tiempos de Galileo, donde este gran científico utilizó lo que se llamó por mucho tiempo, "Método Científico". La investigación científica es un procedimiento que utilizan las personas de ciencias para comprobar hipótesis, solucionar problemas y formular teorías. No hay una investigación científica que sea común para todas las ciencias, pero como se aborda la resolución de problemas va a depender del fenómeno estudiado. Todo investigador o científico debe tener en cuenta en su investigación; detectar el problema, formular hipótesis, tener una recolección de datos para respaldar la hipótesis que lo llevará a la conducción de la solución del problema y aumentará el conocimiento científico que incide en la sociedad (5). La investigación científica es la búsqueda de conocimientos o de soluciones a problemas de carácter científico y cultural, se caracteriza por ser un proceso organizado, objetivo y sistemático (6).

Actualmente han ido en ascenso los procesos investigativos a nivel mundial y los investigadores han enfatizado en publicar sus resultados para elevar el desarrollo científico y compartir el conocimiento, al mismo tiempo para enriquecer sus avales personales y colectivos, de ahí que los procesos académicos y científicos requieran una documentación que avale sus resultados. Las publicaciones científicas se han convertido en un indicador del currículo personal o de una institución como medidor investigativo. Luego del proceso investigativo surge la idea de varias instituciones de contribuir a ese proceso de acreditar resultados, siendo una de las soluciones certificar las publicaciones.

1.1.2 Premios curriculares

Según la Real Academia Española, un premio es una recompensa, galardón o remuneración que se da por algún mérito o servicio (7). Partiendo de esto los autores de la presente investigación plantean que un premio curricular es un reconocimiento a un currículo con determinadas características. Estos pueden ser otorgados a una publicación científica, una persona o un grupo, incluso al resultado de una actividad científica. Los premios en general, y los curriculares en particular, presentan varias características, entre las cuales se pueden señalar que poseen cierta regularidad, un patrocinador que es el encargado de hacer entrega del mismo, el alcance que puede ser interno, nacional o internacional, y poseen cierto valor que puede ser material o no. Actualmente la comunidad universitaria se encuentra enmarcada en la investigación y el desarrollo constante de nuevos productos, servicios y tecnologías, esto trae consigo publicaciones de gran relevancia, productos de *software* y personal altamente calificado, además de impacto social, económico, cultural y científico, implicando que en la universidad la entrega de premios curriculares sea considerada alta dada la participación en las actividades de Ciencia, Tecnología e Innovación de los estudiantes y trabajadores.

1.1.3 Certificación

La certificación es la acción llevada a cabo por una entidad independiente mediante la que se manifiesta que una organización, producto, proceso o servicio, cumple los requisitos definidos en determinadas normas o especificaciones técnicas (8). La Universidad Católica de San Pablo define certificación como un documento que comprueba que una persona u organización cumple con los estándares mínimos para desempeñar una labor en un área determinada. Este documento de certificación es expedido por una organización con reconocido prestigio y experiencia en el área. La certificación es por lo regular voluntaria y válida sólo para la institución donde fue certificada (9).

A partir de los planteamientos anteriores, la certificación a grandes rasgos es emitir un documento que asegure un producto, persona o empresa. Patentizando así la legitimidad y validez del trabajo, así como la autoría de sus creadores. En este proceso interviene una tercera parte que propicia garantía escrita de que un producto, proceso o servicio es acorde con algunos requisitos concretos. En el mismo se identifican tres partes diferenciadas: en primer lugar, el organismo que elabora las normas técnicas que determinan los requisitos específicos que dan pie a la certificación; en segundo lugar, la entidad que emite el documento que demuestra el cumplimiento de dichas normas y en tercer lugar, la entidad verificada.

1.2 Estudio de homólogos

A partir de la necesidad existente en la universidad de desarrollar componentes para gestionar los premios curriculares y certificar las publicaciones, se realizó un estudio de los sistemas que utilizan mecanismos para la gestión de premios y la certificación de publicaciones en el ámbito nacional e internacional. El análisis de estos permite enriquecer y fortalecer la investigación, ya que se pueden identificar las funcionalidades más adecuadas y las mejores prácticas a implementar.

1.2.1 Certificación de publicaciones

A nivel mundial existen varias instituciones que realizan el proceso de certificación de publicaciones científicas, entre ellas se encuentran:

Marco Internacional

Sistema de Certificación de Publicaciones *Web* (publifirma)

El sistema publifirma permite que los usuarios de Firmaprofesional sepan el momento en que una organización publica un documento en su *web*, cuánto tiempo permanece publicado y que el documento no ha sufrido ninguna alteración en el período de su publicación. Dicho de otra manera, publifirma aporta a los clientes de Firmaprofesional[1] la evidencia legal de que un documento se publicó en su *web* en una fecha y a una hora determinada y durante un período concreto de tiempo. El sistema presenta funcionalidades como:

- Registro de publicaciones realizadas, permite visualizar los datos más importantes, dígase, autor, título, descripción y fecha de publicación, con la opción además de descargar la publicación.
- La certificación de publicaciones puede ser manual o automático, ya que posee algoritmos que validan la existencia de la publicación en un período de 5 minutos.
- Emite un certificado digital.

De forma general el sistema solo avala publicaciones de los usuarios de Firmaprofesional, por lo que no es accesible para cualquier persona y no contempla los tipos de publicaciones que se generan en la

[1] Autoridad de Certificación, crea arquitecturas de Certificación, diseña soluciones y servicios basadas en certificación digital

universidad. Sin embargo, este sistema sirvió de base para el estudio de flujo de información que sigue este proceso y funcionalidades de apoyo para la consulta de las publicaciones de los usuarios en la UCI.

Marco Nacional

En Cuba con el avance tecnológico ha crecido gradualmente el índice de investigaciones científicas, a partir de ello los investigadores han encontrado la necesidad de comunicar sus resultados. Muchas instituciones certifican las publicaciones de sus miembros como resultado de medir sus indicadores investigativos, de esta forma facilita la documentación acumulada en expedientes que investigadores, ya sean docentes, trabajadores o especialistas deben presentar para optar por categorías científicas o docentes, premios, distinciones, grados científicos; además avalar o acreditar de cierta forma las publicaciones realizadas. Para el estudio de los sistemas de certificación de publicaciones se tuvo en cuenta las universidades José Martí, Universidad de La Habana, Instituto Superior Politécnico José Antonio Echeverría, Marta Abreu y Universidad de las Ciencias Informática identificándose sistemas homólogos en estas dos últimas.

Certificaciones en Línea. Universidad Central "Marta Abreu" de las Villas (UCVL)

Actualmente esta institución brinda el Servicio de Certificaciones en Línea el que permite a los usuarios efectuar solicitudes para obtener certificaciones de publicación de libros, monografías, artículos de revistas. Está dirigido a todo el personal de la UCLV con el objetivo de validar la existencia de una publicación y que sean añadidas en los expedientes de la universidad.

Los interesados pueden acceder al sistema a través del portal de la biblioteca de la universidad, se autentifican y llenan el formulario de la solicitud. El sistema solo realiza y lista las solicitudes de certificación.

El sistema contempla muy pocas funcionalidades por lo que no es viable para su utilización en la universidad, ya que no realiza un registro y control de las publicaciones y no presenta una interfaz que permita certificar la publicación. El proceso de solicitud no se realiza de forma óptima, pues una vez que se realiza el investigador debe ir personalmente a revisar los datos de su publicación junto a la persona encargada de certificar, implicando que la única utilidad de la petición realizada sea dejar constancia en el sistema.

Tesis Desarrollo del Módulo de Certificación de Publicaciones en Línea para la Universidad de las Ciencias Informáticas

El trabajo de diploma realizado con fin investigativo propone un sistema para la certificación de publicaciones en la universidad, donde los profesores y estudiantes podrán realizar las peticiones de certificación, además tendrán un control de cada publicación certificada que poseen. Por otra parte, los trabajadores designados por la biblioteca podrán tener un mejor control de cada una de las solicitudes realizadas por los usuarios. Este propone funcionalidades que sirven de base a la investigación como:

- Solicitud de certificación.
- Certificación de la publicación, esta funcionalidad permite revisar el listado de solicitudes realizadas, consultar los datos de la solicitud y autoriza o no la emisión del certificado de publicación a nombre del usuario solicitante.
- Consulta de estadísticas, el sistema realiza consultas según los autores más certificados, publicaciones más certificadas, tipos de publicaciones más certificadas, temáticas más certificadas.
- Búsqueda de artículos certificados.

A pesar de que en esta propuesta de solución se definieron funcionalidades indispensables para la certificación de publicaciones en la universidad, no cumple con las necesidades actuales de la DI, ya que no permite la solicitud de impresión del certificado y la gestión de las fuentes de publicación. Sin embargo, es una propuesta de gran ayuda para el desarrollo de la investigación, al ser la mayoría de las funcionalidades mencionadas necesarias para la certificación de publicaciones.

1.2.2 Premios

Así como para acreditar la certificación de publicaciones, para el proceso de control de los premios se identificaron los sistemas que se muestran a continuación:

Marco Internacional

Portal del Ministerio de Educación, Cultura y Deporte del gobierno de España

En el portal se encuentran los servicios de becas, ayudas, subvenciones y premios con los que cuenta este ministerio. La gestión de premios está organizada en las áreas de educación, deporte y cultura en las cuales se contempla las siguientes funcionalidades:

- Lista de convocatorias al premio.
- Especificaciones para optar por el premio.
- Lista de resultados al premio, se encuentran distribuidos por premio y año.

Este portal es mayormente informativo ya que solo muestra información a sus usuarios de las convocatorias y resultados de los premios. Estos elementos informativos y la estructura que muestra aportaron nuevos elementos a la investigación, así como la forma en que se encuentran organizados los datos. A pesar de que no posee funcionalidades consideradas críticas para el desarrollo de la solución como:

- Solicitudes de un premio
- Evaluación de las propuestas al premio.

Premio odebrecht

Portal que gestiona los procesos del premio odebrecht[2] para el desarrollo sostenible, con el fin de incentivar a jóvenes universitarios a pensar y desarrollar soluciones de ingeniería desde una perspectiva de sostenibilidad. En la información que muestra se especifican los criterios de participación, prohibiciones, inscripciones, criterios de calificación y premiación, que constituyó una base para el desarrollo de sistema, ya que es necesario tener en cuenta como se realiza la gestión de:

- Criterios de evaluación.
- Solicitudes.

[2] **Odebrecht** es una Organización global de origen brasileño que realiza negocios diversificados en todo el mundo. Su actividad se sustenta en tres grandes áreas de negocios: Ingeniería y Construcción, Inversiones en Infraestructura y Energía, e Industria.

- Elección de premiados.

El sitio permite a todos los interesados a optar por el premio a:

- Realizar la solicitud de participación.
- Subir la solución informática a evaluar.
- Consultar los premios otorgados en años anteriores.

Funcionalidades que se tuvieron en cuenta de forma conceptual para la realización de componente. Aunque el portal está enfocado solo a un premio en específico y no contaba con las funcionalidades necesarias para realizar el proceso de calificación para elegir los premiados.

SERES

El sitio *web* gestiona la entrega del premio SERES[3], que promueve el compromiso de las empresas en la mejora de la sociedad con actuaciones responsables alineadas con la estrategia de la compañía y generando valor para todos. Este muestra a sus usuarios la convocatoria al premio, los premiados en años anteriores, así como un formulario para realizar la solicitud. El estudio y análisis de estas funcionalidades generaron los siguientes elementos a tener en cuenta para el desarrollo de la solución:

- La estructura e información a tener en cuenta para realizar la solicitud.
- Elementos de búsqueda para consultar los premios.

Este portal no es viable para su utilización en la universidad ya que solo tiene en cuenta el premio SERES, el cual no es un premio curricular por lo que se haría muy compleja su adaptación a los nuevos requerimientos, además de no presentar funcionalidades para realizar la evaluación de las propuestas.

1.2.3 Análisis de los sistemas estudiados

Para el desarrollo de la solución se tuvo en cuenta el estudio de otros sistemas según los requerimientos de los clientes y las necesidades del negocio. A partir de la investigación realizada se analizó la información

[3] Fundación SERES es una organización no gubernamental la cual tiene cabida todas las empresas y organizaciones con independencia de su ubicación y tamaño.

manejada para la entrega de premios y certificación de publicaciones y se identificaron funcionalidades que sirvieron de apoyo para la implementación de los componentes como se muestra a continuación:

- El registro de certificados en el sistema publifirma permite realizar búsquedas especializadas por nombre, fecha y autor de la publicación.
- La solicitud en el portal SERES la mayoría de los campos a llenar son de texto largo, ya que se requiere redactar una autovaloración para concursar por el premio.
- La consulta de premios en el portal odebrecht permite realizar búsquedas especializadas por fecha y premiados.

Actualmente las tecnologías utilizadas por estos sistemas son incompatibles con el marco de trabajo GUUD, que es utilizado por el SGU donde serán integrados los componentes a desarrollar. Para la utilización de alguno de estos sistemas sería necesario añadir nuevas funcionalidades y cambiar las tecnologías de ser posible, lo cual tomaría demasiado tiempo por lo que no se considera una solución viable. Además de no tenerse acceso al código fuente de la mayoría de los sistemas.

1.3 Metodología de desarrollo de *software*

Los componentes Publicaciones y Premios serán desarrollados utilizando las tecnologías, metodologías y herramientas del Sistema de Gestión Universitaria al que serán integrados, establecidas por el departamento de desarrollo de la DIN.

1.3.1 Metodología

Metodología puede entenderse como el conjunto de procedimientos que determinan una investigación de tipo científico o marcan el rumbo de una exposición doctrinal. Depende de los postulados que el investigador crea que son válidos, ya que la acción metodológica será su herramienta para analizar la realidad estudiada. La metodología para ser eficiente debe ser disciplinada, sistemática y permitir un enfoque que permite analizar un problema en su totalidad (10). Las metodologías se pueden identificar en dos tipos: ágiles y robustas o tradicionales, entre las ágiles se destacan: *eXtreme Programming (XP), SCRUM, Crystal Methodologies* y entre las robustas o tradicionales*: Microsoft Solution Frameworks (MSF), Métrica 3,* Proceso Unificado de Desarrollo (*RUP*, por sus siglas en inglés).

Desarrollo Ágil con Calidad (DAC)

La metodología a utilizar es DAC, se basa en un proceso de desarrollo de *software* que combina las metas y prácticas de las áreas de procesos del nivel dos de CMMI (*Capability Maturity Model Integration*) con las buenas prácticas de la dirección y desarrollo ágil de proyectos de *software*. Es un proceso colaborativo, recursivo-iterativo, además es incremental y guiado por procesos y requisitos. Su modelo del proceso es una adaptación del modelo en Cascada a los modelos Programación Extrema y Desarrollo Concurrente. Está enfocado a proyectos pequeños o grandes divididos en sub-proyectos que desarrollan *software* de gestión basado en componentes (11).

DAC plantea que el problema una vez identificado y definido debe ser descompuesto en problemas más pequeños, y si es necesario, realizar con estos la misma operación. Cada sub-problema será resuelto mediante un componente y el problema resuelto será el *software* o producto final; por lo que las entregas en DAC son a nivel de iteración, en la que habrá obligatoriamente un incremento del producto a partir de la solución de un componente del mismo. Además en cada iteración se define como mínimo un hito a cumplir por cada fase del proceso. Al finalizar cada iteración se realiza la integración del componente al producto obtenido hasta el momento realizando pruebas de integración. Al finalizar las iteraciones se pueden realizar liberaciones del componente y transición del mismo dentro de la fase Cierre de iteración. Las iteraciones no tienen que desarrollarse todas al mismo tiempo sino que al contar con un equipo pequeño este se va a ir moviendo de una iteración a otra a medida que estas vayan terminando de acuerdo a un orden de prioridad establecido en el plan del proyecto (11).

Este proceso tiene 8 actividades del marco de trabajo del proceso común, llamadas: fases o procesos del ciclo de vida: inicio, análisis, diseño arquitectónico, requisitos, construcción, cierre de iteración (opcional), liberación, transición y cierre, ocurriendo las iteraciones concurrentes entre las fases de requisitos, construcción y cierre de iteración. Además, entre las fases de requisitos y construcción puede ocurrir un ciclo pues a medida que los requisitos son especificados estos pueden ir entrando a la fase de construcción. El proceso tiene también dos áreas de procesos de protección: gestión de proyectos y soporte así como dos fases o procesos horizontales cuyas tareas están presentes en varias de las fases del proceso común en forma de subprocesos: arquitectura y planificación (11).

1.4 Herramientas y tecnologías para el desarrollo de la aplicación

En la presente investigación se utiliza para el desarrollo de los componentes el marco de trabajo GUUD (acrónimo creado con las iniciales de los departamentos de la DIN: Gestión Universitaria, Universidad Digital y Gestión Documental), el cual surge de la integración del marco de trabajo *CodeIgniter* con *JQuery*.

CodeIgniter

Es un entorno de desarrollo abierto que permite crear *webs* dinámicas con PHP (*Hypertext Pre-Processor*). Su principal objetivo es ayudar a que los desarrolladores, puedan realizar proyectos mucho más rápido que creando toda la estructura desde cero, proveyendo un rico juego de librerías para tareas comúnmente necesarias, así como una interfaz simple y estructura lógica para acceder a esas librerías. Permite concentrase en el desarrollo del proyecto en cuestión, minimizando la cantidad de código necesaria para realizar las tareas. *CodeIgniter* usa el patrón de diseño arquitectónico Modelo-Vista-Controlador como paradigma de arquitectura de desarrollo, el cual se separa en 3 capas distintas: la representación de datos, el interfaz de usuario y el controlador de eventos respectivamente (13).

JQuery

Es un *software* libre y de código abierto. Posee un doble licenciamiento bajo la Licencia MIT y la Licencia Pública General de GNU v2, permitiendo su uso en proyectos libres y privativos. Ofrece una serie de funcionalidades basadas en *JavaScript* que de otra manera requerirían de mucho más código. Es decir, con las funciones propias de esta librería se logran grandes resultados en menos tiempo y espacio. Es una biblioteca o *Framework* de *JavaScript* (14).

1.4.1 Marco de trabajo. GUUD 2.0

El marco de trabajo GUUD incorpora novedades y modificaciones en su infraestructura, a continuación se listan las mismas.

Del lado del cliente:

1. Se implementaron una serie de *widgets* para utilizarlos de interfaz de algunos de los *widgets* base de jquery-ui.
2. Se le implementó un *plugin* a *jQuery* para el manejo de espacios de nombre e internacionalización.

3. Se implementaron funciones comunes para todo el sistema (contenidas en los archivos *core.js y* common.*js*).

Del lado del servidor (hechas con *CodeIgniter*):

1. Se le agregó el manejo de excepciones y mensajes.
2. Se le implementó el *IoC*[4] para la interacción entre componentes.
3. Se le añadió la característica de la modularidad o sea que una aplicación pueda dividirse en componentes. *CodeIgniter* no cuenta con esta posibilidad.
4. Se añadieron, modificaron y extendieron los *helpers* o asistentes.

Cada petición es recibida por el controlador frontal (*index.php*) que inicializa todos los recursos que necesita el marco de trabajo para procesar la petición, el enrutador examina la solicitud HTTP (Protocolo de transferencia de hipertexto) para determinar qué debería hacer con esta.

Si existe este archivo de caché, se lo envía directamente al navegador, sin pasar por la ejecución normal del sistema. Antes de cargar el controlador, por razones de seguridad, se filtra la solicitud HTTP y cualquier otro dato enviado por el usuario. El controlador carga las librerías (que funcionan como intermediarias entre las capas de negocio y las de acceso a datos), las bibliotecas del núcleo, *helpers* y cualquier otro recurso requerido para procesar una solicitud, además de procesar las vistas (estas a su vez obtienen información de los *scripts*). Si la caché está habilitada la vista se cachea para que futuras peticiones que la necesiten puedan ser servidas.

1.4.2 Herramienta Visual Paradigm para UML v 5.0

Es una herramienta de Lenguaje Unificado de Modelado (UML, por sus siglas en inglés) profesional que soporta el ciclo completo de desarrollo del *software* a través de la representación de todo tipo de diagramas. Propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación. Constituye una herramienta privada disponible en varias ediciones, cada una destinada a las necesidades: *Enterprise, Professional, Community, Standard, Modeler* y *Personal*. Existe una alternativa

[4] **IoC** *Inversion of Control* (Inversión de Control)

libre y gratuita de este *software*, la versión Visual Paradigm UML 8.0 *Community Edition*. Fue diseñado para una amplia gama de usuarios interesados en la construcción de sistemas de *software* de forma fiable, como se evidencia al hacer diagramas el enfoque orientado a objetos (12).

Características de Visual Paradigm

- ✓ Disponibilidad en múltiples plataformas como Windows y Linux.
- ✓ Diseño centrado en casos de uso y enfocado al negocio que generan un *software* de mayor calidad.
- ✓ Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- ✓ Licencia gratuita y comercial.
- ✓ Soporta aplicaciones *web*.
- ✓ Generación de bases de datos como transformación de diagramas de Entidad-Relación en tablas de base de datos.
- ✓ Editor de figuras.

1.4.3 **Entorno de desarrollo.** NetBeans 8.0

NetBeans es un proyecto exitoso de código abierto con una gran base de usuarios, una comunidad en constante crecimiento. *NetBeans* IDE (Entorno de Desarrollo Integrado) es una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java, pero puede servir para cualquier otro lenguaje de programación. Existe además un número importante de componentes para extender el *NetBeans* IDE. Este es un producto libre y gratuito sin restricciones de uso. Ofrece una versión del IDE hecho a medida para el desarrollo de sitios *web* en PHP que comprenden una variedad de secuencias de comandos y lenguajes de marcado; se integra dinámicamente con HTML, *JavaScript* y CSS (15).

1.4.4 **Administrador de base de datos.** PgAdmin III

PgAdmin III es una aplicación gráfica para administrar el gestor de bases de datos *PostgreSQL*. Es capaz de gestionar versiones a partir de *PostgreSQL* 7.3 ejecutándose en cualquier plataforma. Está diseñado para responder a las necesidades de todos los usuarios, desde escribir consultas SQL (Lenguaje de Consulta Estructurado) simples hasta desarrollar bases de datos complejas. La interfaz gráfica soporta

todas las características de *PostgreSQL* y facilita enormemente la administración. La aplicación también incluye un editor SQL con resaltado de sintaxis, un editor de código de la parte del servidor y un agente para lanzar scripts programados. La conexión al servidor puede hacerse mediante conexión TCP/IP[5] y puede encriptarse mediante SSL[6] para mayor seguridad. Está soportado por la licencia BSD[7] (16).

1.4.5 **Sistema gestor de base de datos**. PostgreSQL 9.4.1

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD y con su código fuente disponible libremente. Es el sistema de gestión de bases de datos de código abierto más potente del mercado y en sus últimas versiones no tiene nada que envidiarle a otras bases de datos comerciales. *PostgreSQL* utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando (17).

1.4.6 **Servidor *web*.** Apache 2.4.7

Apache es el servidor *web* hecho por excelencia, su configuración, robustez y estabilidad hacen que cada vez millones de servidores reiteren su confianza en este programa. La licencia Apache permite hacer lo que se quiera con el código fuente siempre que les reconozcas su trabajo.

- Corre en una multitud de sistemas operativos, lo que lo hace prácticamente universal.
- Es una tecnología gratuita de código abierto.
- Es un servidor altamente configurable de diseño modular, al que es muy fácil ampliar sus capacidades.
- Trabaja con gran cantidad de lenguajes como *Perl*, PHP y otros lenguajes de script.
- Permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor.

[5] **TCP/IP** es un conjunto de protocolos de red que permiten la trasmisión de datos entre computadoras

[6] **SSL** (***Secure Sockets Layer*)** o Capa de conexión segura es un protocolo criptográfico que proporciona comunicaciones seguras por una red

[7] **BSD** Licencia de *Software* Libre Permisiva

Tiene una alta configurabilidad en la creación y gestión de registros. Permite la creación de ficheros de registro a medida del administrador, de este modo se puede tener un mayor control sobre lo que sucede en el servidor (18).

1.4.7 **Prototipos de interfaz.** Evolus Pencil 1.3.4

Evolus Pencil fue creado basándose en la tecnología de Mozilla, es una herramienta gratuita y de código abierto para diseñar tus prototipos *web*. Es de gran ayuda para diseñadores y desarrolladores *web* que permite diseñar rápida y fácilmente documentos de propuesta para clientes (19).

Características de *Evolus Pencil*.

- Diseño de plantillas y creación de prototipos.
- Documento de múltiples páginas.
- Vínculos y enlaces internos entre las páginas.
- Edición de texto enriquecido.
- Instalación de plantillas definidas por el usuario.
- Operaciones estándar de dibujo: alineación, z-orden, escalado, rotación, dimensiones, entre otros.
- Acciones para objetos.
- La exportación a HTML[8], PNG[9], documento de *Openoffice.org*, documento de *Word* y *PDF*[10].

[8] **HTML** lenguaje de marcado de hipertexto

[9] **PNG** (siglas en inglés de Gráficos de Red Portátiles, pronunciadas "ping") formato gráfico basado en un algoritmo de compresión sin pérdida para *bitmaps* no sujeto a patentes.

[10] **PDF** (sigla del inglés *Portable Document Format*, “formato de documento portátil”) es un formato de almacenamiento para documentos digitales independiente de plataformas de *software* o *hardware*.

1.5 Lenguajes utilizados para el desarrollo de la aplicación

1.5.1 **Lenguaje unificado de modelado.** UML 2.0

El Lenguaje Unificado de Modelado se define como un lenguaje que permite especificar, visualizar y construir los artefactos de los sistemas de *software*. Es el más conocido y utilizado en la actualidad. Se usa para entender, diseñar, hojear, configurar, mantener y controlar la información sobre tales sistemas. Está pensado para usarse con todos los métodos de desarrollo, etapas del ciclo de vida, dominios de aplicación y medios. UML incluye conceptos semánticos, notación y principios generales. Se puede aplicar en el desarrollo de *software* entregando gran variedad de formas para dar soporte a una metodología de desarrollo, pero no especifica en sí mismo qué metodología o proceso usar (20).

Un modelo UML está compuesto por tres clases de bloques de construcción:

- Elementos: son abstracciones de cosas reales o ficticias (objetos y acciones).
- Relaciones: relacionan los elementos entre sí.
- Diagramas: son colecciones de elementos con sus relaciones.

1.5.2 **Lenguaje de programación**. PHP 5.6.7

Es un lenguaje de *script* interpretado en el lado del servidor utilizado para la generación de páginas *web* dinámicas, similar al ASP de *Microsoft* o el JSP de *Sun*, embebido en páginas HTML y ejecutado en el servidor. La mayor parte de su sintaxis ha sido tomada de C, Java y Perl con algunas características específicas de sí mismo. La meta del lenguaje es permitir rápidamente a los desarrolladores la generación dinámica de páginas. No es un lenguaje de marcas como podría ser HTML, XML[11] o WML[12] (21).

Ventajas:

- Lenguaje multiplataforma.
- Completamente orientado al desarrollo de aplicaciones *web* dinámicas con acceso a información almacenada en una base de datos.

[11] **XML** Lenguaje de marca extensible

[12] **WML** Lenguaje de marcado inalámbrico

- Capacidad de conexión con la mayoría de los motores de bases de datos que se utilizan en la actualidad, destaca su conectividad con *MySQL* y *PostgreSQL*.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Permite aplicar técnicas de programación orientada a objetos.
- No requiere definición de tipos de variables aunque sus variables se pueden evaluar también por el tipo que estén manejando en tiempo de ejecución.

1.5.3 **Lenguaje de programación**. JavaScript 1.8

Es un lenguaje de programación interpretado. No requiere de compilación ya que funciona del lado del cliente, los navegadores son los encargados de interpretar estos códigos.

- Maneja objetos dentro de la página *web* y sobre ese objeto se pueden definir diferentes eventos. Dichos objetos facilitan la programación de páginas interactivas, a la vez que se evita la posibilidad de ejecutar comandos que puedan ser peligrosos para la máquina del usuario, tales como formateo de unidades y modificar archivos.
- Es dinámico, responde a eventos en tiempo real. Eventos como presionar un botón, pasar el puntero del *mouse* sobre un determinado texto o el simple hecho de cargar la página o caducar un tiempo. Con esto se puede cambiar totalmente el aspecto de la página al gusto del usuario, evitando tener en el servidor una página para cada gusto, hacer cálculos en base a variables cuyo valor es determinado por el usuario (22).

1.5.4 **Lenguaje de Marcado de Hipertexto.** HTML 4

Es un lenguaje de marcas orientado a la publicación de documentos en la *web*. La mayoría de las marcas son semánticas. Es un lenguaje extensible, al que se le pueden añadir nuevas características, marcas y funciones. Los documentos HTML están formados por una serie de bloques de texto con una entidad lógica (titulares, párrafos y listas). La interpretación de estas entidades se deja al navegador, lo cual da una gran flexibilidad a la presentación del documento, que puede ser mostrado en terminales gráficos o de texto. El HTML, es el que permite diseñar los hipertextos. Actualmente la mayoría de los procesadores de textos disponen de opciones para guardar los documentos en este formato, por lo que no presenta dificultad (23).

1.5.5 **Lenguaje de marcas extensible.** XML 1.0

El lenguaje de marcas extensible es un metalenguaje que permite definir lenguajes de marcado adecuados a usos específicos. Aunque a primera vista un documento XML puede parecer similar a HTML hay una diferencia fundamental: un documento XML contiene datos que se auto-definen, o sea no posee etiquetas prefijadas con anterioridad, ya que es el propio diseñador el que las crea, dependiendo del contenido del documento. En XML se separa el contenido de la presentación de forma total. Entre sus ventajas se encuentra su aceptación casi universal, su legibilidad y su carácter auto-contenido. Al ser el tamaño de los documentos XML mayor que el de sus equivalentes binarios, su procesamiento requiere más recursos, por lo que no resulta adecuado en aplicaciones en las que la eficiencia sea un objetivo prioritario. El mismo permite representar datos de forma homogénea en entornos heterogéneos, lo que facilita la interoperabilidad entre distintos sistemas (24).

1.5.6 **Lenguaje de consulta estructurado.** SQL: 2008

El *Structured Query Language* (SQL) es el lenguaje estándar ANSI/ISO de definición, manipulación y control de bases de datos relacionales. Es un lenguaje declarativo: sólo hay que indicar qué se desea hacer. En cambio, en los lenguajes procedimentales es necesario especificar cómo hay que hacer cualquier acción sobre la base de datos. El SQL es muy parecido al lenguaje natural; concretamente, se parece al inglés y es muy expresivo. Por estas razones y como lenguaje estándar, el SQL es un lenguaje con el que se puede acceder a todos los sistemas relacionales (25).

1.5.7 **Hojas de Estilo en Cascada.** CSS 3

Las Hojas de Estilo en Cascada (CSS por sus siglas en inglés) hacen referencia a un lenguaje usado para describir la presentación semántica (el aspecto y formato) de un documento escrito en lenguaje de marcas. Su aplicación más común es dar estilo a páginas *webs* escritas en lenguaje HTML. Las Hojas de Estilo en Cascada en su versión 3 están divididas en varios documentos separados, llamados "módulos". Cada módulo añade nuevas funcionalidades a las definidas en CSS 2, de manera que se preservan las anteriores para mantener la compatibilidad (26).

1.6 Conclusiones parciales

Con el análisis de los conceptos relacionados con la investigación, se logró comprender los elementos teóricos que respaldan el presente trabajo. El estudio de sistemas utilizados en la actualidad para los procesos de premios y certificación de publicaciones, en el mundo, en el país y en la universidad; permitió adquirir un conjunto de conocimientos previos sobre sus características fundamentales. La caracterización de las herramientas, metodologías y lenguajes definidos por la DIN, permitió la familiarización con los elementos del entorno de desarrollo y la obtención de los conocimientos necesarios sobre sus características para utilizarlos en la construcción de la propuesta de solución.

Capítulo 2: Descripción y análisis de la solución propuesta

En este capítulo se presenta la propuesta de solución al problema planteado en el diseño teórico de la investigación. Se desarrolla el modelado del proceso de negocio para una mayor comprensión del entorno de investigación y se describen los conceptos fundamentales. Se especifican las técnicas de obtención de requisitos, identificando así los requisitos funcionales y no funcionales de los componentes Publicaciones y Premios. Además se realiza la descripción de la arquitectura, los patrones empleados y el diagrama de despliegue.

2.1 Modelado de proceso de negocio

Un modelado de procesos de negocio es un conjunto estructurado y medible de actividades diseñadas para producir un producto especificado para un cliente o mercado específico. Implica un fuerte énfasis en cómo se ejecuta el trabajo dentro de la organización, en contraste con el énfasis en el qué, característico de la focalización en el producto (27).

2.1.1 Flujo actual del proceso de certificación de publicaciones

El proceso comienza cuando el investigador envía un correo electrónico de solicitud con los datos personales y la fuente de donde proviene la publicación, dígase por fuente: monografía, libro, revista, artículo y memoria de evento. Posteriormente se verifica la existencia de la publicación según la fuente referenciada, a partir de este análisis, de encontrarse la publicación se confecciona un certificado que acredite la validez de la publicación. Este certificado es enviado al departamento de impresión y luego es firmado por el director de la DI. Finalmente estos certificados son entregados al investigador personalmente en la biblioteca de la universidad. En caso de no encontrarse la publicación se le notifica al investigador mediante correo electrónico, especificando el motivo del rechazo.

2.1.2 Reglas de negocio de certificación de publicaciones

1. Se entrega un certificado a cada autor de la publicación.
2. El investigador sólo podrá solicitar el certificado una vez para cada publicación.
3. Las publicaciones están compuestas por un autor principal y los demás miembros se consideran coautores.

4. Los coautores se definen según su nivel de participación en la publicación.
5. Sólo se podrá solicitar una certificación de publicaciones cuando al menos uno de los investigadores pertenezca a la universidad.

Para mejor comprensión del proceso de negocio dirigirse al Anexo 2: Diagrama de proceso de negocio certificación de publicaciones.

2.1.3 Flujo actual del proceso de premios

Para cada nueva edición de un premio la DI envía la convocatoria a las distintas áreas, donde se encuentran definidas las categorías y criterios de evaluación por los cuales se regirá la selección de los premiados. En cada área se seleccionan las propuestas y se confecciona un expediente en el que se recogen todos los datos necesarios para demostrar que el investigador cumple con las condiciones para optar por el premio. Los expedientes deben ser enviados a la DI en la fecha límite que se haya definido donde serán analizados y rechazados en caso que presenten ausencia de datos o no se encuentren correctamente confeccionados.

La DI confecciona una comisión para cada categoría, compuesta como mínimo por un presidente. Para elegir los premiados la comisión se reúne en un período definido donde valoran las propuestas teniendo en cuenta cada criterio de evaluación, para posteriormente premiar las mejores propuestas. Además, los premios pueden ser otorgados por instituciones del MES, en estos casos la DI selecciona las propuestas que representarán a la universidad y envía los expedientes.

Para mejor comprensión del proceso de negocio dirigirse al Anexo 3: Diagrama de proceso de negocio de premios.

2.1.4 Reglas de negocio de premios

1. Se confeccionará un expediente por cada propuesta al premio de las distintas áreas.
2. El expediente sólo será admitido en el plazo de entrega definido en la edición del premio.
3. Si el premio no es convocado por la universidad, el proceso de elección de los premiados será realizado por la institución promotora.
4. De ser convocado el premio por la universidad, este será evaluado por el consejo científico de la universidad.

5. Entre los miembros del consejo científico se elegirá un presidente el cual debe pertenecer a la universidad.
6. Cada criterio de evaluación tendrá asociado un peso dependiendo de la importancia de cada criterio.
7. Cada miembro del consejo científico tendrá asociado un peso dependiendo de la importancia de la evaluación de cada uno.
8. El presidente del consejo científico será el responsable de decidir los premiados luego de realizada la evaluación.

2.2 Requisitos del sistema

Los requisitos para un sistema de *software* determinan lo que debe hacer el sistema y definen las restricciones en su funcionamiento e implementación, es decir, lo que el *software* debe hacer y bajo qué circunstancias debe hacerlo (28).

2.2.1 Técnicas para la obtención de requisitos

El descubrimiento de requisitos es el proceso de recoger información sobre el sistema propuesto y los existentes, extraer los requisitos del usuario y del sistema de esta información (29). La ingeniería de requisitos ofrece algunas técnicas que se utilizan para identificar y extraer los requisitos, dentro de las que se encuentran: puntos de vista, escenarios, casos de uso, prototipos, tormenta de ideas, entre otros. De estas técnicas se utilizaron en la investigación las siguientes:

Sesiones de tormentas de ideas (*Brainstorming*): se realizaron reuniones con la participación del cliente e integrantes del proyecto, con el objetivo de generar ideas respecto a las funcionalidades e integración de los premios curriculares y la certificación de publicaciones con el sistema. Esto ayudó a definir nuevas funcionalidades con el apoyo y críticas de los participantes en las reuniones.

Entrevistas: se realizaron entrevistas al cliente, posibilitando el intercambio con los expertos en el área a tratar, definir ideas puntuales y nuevos aspectos que no se contemplaban en el proceso actual. Para ver las entrevistas dirigirse al Anexo 1.

Prototipos: permite obtener una pequeña muestra de lo que sería el producto final. Se diseñaron un grupo de prototipos desechables que se mostraron al cliente, con el objetivo de mostrar de manera gráfica el diseño del sistema y rectificar algunos aspectos antes de desarrollar los componentes.

2.2.2 Requisitos funcionales

Los requisitos funcionales de un sistema describen que debe hacer el sistema. Estos requisitos dependen del tipo de *software* que se desarrolle, de los posibles usuarios y del enfoque general tomado por la organización al redactar el requisito. Los requisitos funcionales del sistema describen con detalle la función de este, sus entradas, salidas y excepciones (29). En la Tabla 1 se muestran los requisitos funcionales que debe cumplir el componente Publicaciones y en la Tabla 2 los del componente Premios.

A continuación se muestran las Tablas 1 y 2 con la lista de requisitos funcionales:

Tabla 1: Requisitos de Publicaciones

No.	Requisitos	Prioridad	Complejidad
RFPU[13]1	Crear revista	Alta	Alta
RFPU2	Crear libro	Alta	Alta
RFPU3	Crear monografía	Alta	Alta
RFPU4	Crear otras fuentes	Alta	Alta
RFPU5	Modificar revista	Media	Media
RFPU6	Modificar libro	Media	Media
RFPU7	Modificar monografía	Media	Media
RFPU8	Modificar otras fuentes	Media	Media
RFPU9	Eliminar fuente	Baja	Baja
RFPU10	Listar fuentes	Media	Baja
RFPU11	Ver detalles de fuente	Baja	Baja
RFPU12	Listar fuentes a analizar	Media	Baja

[13] **RFPU** Requisito funcional Publicaciones

RFPU13	Ver detalles de una fuente a analizar	Baja	Baja
RFPU14	Analizar revista	Alta	Media
RFPU15	Analizar libro	Alta	Media
RFPU16	Analizar monografía	Alta	Media
RFPU17	Analizar otras fuentes	Alta	Media
RFPU18	Crear solicitud de certificación	Alta	Alta
RFPU19	Modificar solicitud de certificación	Media	Media
RFPU20	Eliminar solicitud de certificación	Baja	Baja
RFPU21	Ver detalles de solicitud de certificación	Baja	Baja
RFPU22	Listar solicitudes de certificación	Media	Baja
RFPU23	Ver detalles de solicitud de certificación a analizar	Baja	Baja
RFPU24	Analizar solicitud de certificación	Alta	Media
RFPU25	Listar solicitudes de certificación a analizar	Media	Baja
RFPU26	Notificar certificación realizada	Media	Media
RFPU27	Listar mis publicaciones	Media	Baja
RFPU28	Ver detalles de mis publicaciones	Media	Baja
RFPU29	Crear certificado de publicación	Alta	Alta
RFPU30	Solicitar Impresión	Media	Baja
RFPU31	Listar otras publicaciones	Media	Baja
RFPU32	Ver detalles de otras publicaciones	Media	Baja
RFPU33	Listar certificados por imprimir	Media	Baja
RFPU34	Imprimir certificado	Media	Media
RFPU35	Ver detalles de certificado por imprimir	Media	Baja
RFPU36	Listar certificados impresos	Media	Baja
RFPU37	Ver detalles de certificados impresos	Media	Baja
RFPU38	Registrar editorial	Media	Baja

RFPU39	Modificar editorial	Media	Baja
RFPU40	Ver detalles de editorial	Media	Baja
RFPU41	Listar editorial	Baja	Baja
RFPU42	Crear frecuencia	Media	Baja
RFPU43	Modificar frecuencia	Media	Baja
RFPU44	Ver detalles de frecuencia	Media	Baja
RFPU45	Listar frecuencias	Baja	Baja
RFPU46	Registrar referencia	Media	Baja
RFPU47	Modificar referencia	Media	Baja
RFPU48	Ver detalles de referencia	Media	Baja
RFPU49	Listar referencia	Baja	Baja
RFPU50	Registrar temática	Media	Baja
RFPU51	Modificar temática	Media	Baja
RFPU52	Ver detalles de temática	Media	Baja
RFPU53	Listar temáticas	Baja	Baja
RFPU54	Crear tipo de código	Media	Baja
RFPU55	Modificar tipo de código	Media	Baja
RFPU56	Ver detalles tipo de código	Media	Baja
RFPU57	Listar tipos de códigos	Baja	Baja
RFPU58	Crear idioma	Media	Baja
RFPU59	Modificar idioma	Media	Baja
RFPU60	Ver detalles de un idioma	Media	Baja
RFPU61	Listar idiomas	Baja	Baja
RFPU62	Listar resolución	Baja	Baja
RFPU63	Crear resolución	Media	Media
RFPU64	Modificar resolución	Media	Media

RFPU65	Ver detalles resolución	Baja	Baja

Tabla 2: Requisitos de Premios.

No	Nombre	Prioridad	Complejidad
RFPR[14]1	Crear premio	Alta	Alta
RFPR2	Modificar premio	Media	Media
RFPR3	Listar premios	Baja	Baja
RFPR4	Ver detalles de un premio	Baja	Baja
RFPR5	Crear institución	Media	Media
RFPR6	Modificar institución	Media	Media
RFPR7	Listar instituciones	Media	Baja
RFPR8	Ver detalles de institución	Baja	Baja
RFPR9	Crear edición de un premio	Alta	Alta
RFPR10	Modificar edición de un premio	Media	Alta
RFPR11	Listar ediciones de un premio	Media	Baja
RFPR12	Ver detalles de una edición de un premio.	Baja	Baja
RFPR13	Crear categoría	Media	Baja
RFPR14	Modificar categoría	Media	Baja
RFPR15	Listar categorías	Media	Baja
RFPR16	Ver detalles de categoría	Baja	Baja
RFPR17	Crear criterio de evaluación	Media	Baja
RFPR18	Modificar criterio de evaluación	Media	Baja
RFPR19	Ver detalles de criterio de evaluación	Media	Baja
RFPR20	Listar criterios de evaluación	Baja	Baja

[14] **RFPR** Requisito funcional Premios

RFPR21	Crear comisiones de evaluación	Alta	Media
RFPR22	Modificar comisiones de evaluación	Media	Media
RFPR23	Listar comisiones de evaluación	Media	Baja
RFPR24	Ver detalles de una comisión de evaluación	Baja	Baja
RFPR25	Solicitar participación en premio	Alta	Alta
RFPR26	Modificar solicitud de participación en premio	Alta	Alta
RFPR27	Eliminar solicitud de participación en premio	Media	Baja
RFPR28	Listar solicitudes de premios	Media	Media
RFPR29	Ver detalles de una solicitud de un premio	Baja	Baja
RFPR30	Analizar solicitud de un premio.	Alta	Media
RFPR31	Listar solicitudes de premio por analizar	Media	Media
RFPR32	Listar premiados	Media	Media
RFPR33	Listar mis premios	Baja	Media
RFPR34	Ver detalles de mis premios	Baja	Baja
RFPR35	Listar otros premios	Baja	Media
RFPR36	Ver detalles de otros premios	Baja	Baja
RFPR37	Crear alcance	Media	Baja
RFPR38	Modificar alcance	Media	Baja
RFPR39	Listar alcance	Media	Baja
RFPR40	Ver detalles de alcance	Baja	Baja
RFPR41	Crear estado	Media	Baja
RFPR42	Modificar estado	Media	Baja
RFPR43	Listar estados	Media	Baja
RFPR44	Ver detalles de estado	Baja	Baja
RFPR45	Crear planilla de solicitud	Alta	Alta

RFPR46	Modificar planilla de solicitud	Alta	Alta
RFPR47	Listar planilla de solicitud	Baja	Baja
RFPR48	Ver detalles de planilla de solicitud	Baja	Baja
RFPR49	Crear atributo	Alta	Media
RFPR50	Modificar atributo	Alta	Media
RFPR51	Listar atributo	Baja	Baja
RFPR52	Ver detalles atributo	Baja	Baja
RFPR53	Evaluar propuesta	Media	Media
RFPR54	Listar propuestas	Media	Media
RFPR55	Ver detalles propuestas	Baja	Baja
RFPR56	Crear expediente	Alta	Alta
RFPR57	Elegir premiados	Media	Media

2.2.3 Requisitos no funcionales

Los requisitos no funcionales son aquellos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de este como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento. Define las restricciones del sistema como la capacidad de los dispositivos de entrada/salida y las representaciones de datos que se utilizan en las interfaces del sistema (28). Para la implementación de los componentes se definieron los siguientes requisitos no funcionales.

Tabla 3: Requisitos no funcionales

Usabilidad
RnF[15] 1. El sistema debe presentar un menú lateral y una barra de iconos flotantes que permitan el acceso rápido a la información por parte de los usuarios.
RnF 2. Las vistas del sistema deben indicar en cada momento la acción que se está realizando, así como los íconos deben estar representados por una imagen acorde a la acción que se realiza.

[15] Requisito no funcional

RnF 3. Sólo se mostrarán a los usuarios aquellas acciones o informaciones del menú lateral a las que por su responsabilidad o rol dentro del negocio necesiten acceder.
Software
RnF 4. Para el despliegue del sistema se debe contar en el servidor de bases de datos con *PostgreSQL* 9.4, bajo el sistema operativo *CentOS* 6.6 o superior.
RnF 5. Para el despliegue del sistema se debe contar en el servidor de aplicaciones *web* con: PHP v 5.6 con las librerías php5-ldap, php5-gd, php5-mcrypt, php5-pgsql, php5-xsl, php5-openssl; Apache 2.2 con el componente *rewrite* activado y JRE[16] 6 o superior.
RnF 6. Para el uso del sistema se recomienda como mínimo una PC cliente con el navegador *web* Mozilla Firefox 26.0.
Confiabilidad
RnF 7. El tratamiento de las excepciones permitirá un seguimiento hasta guardar información, acerca del lugar dónde se produjo el error y de los parámetros utilizados en el sistema que lo provocaron. Cuando ocurre una excepción el sistema mostrará un mensaje explicativo del error ocurrido.
RnF 8. El sistema puede permanecer inactivo durante 10 minutos. Al cumplirse este término se cerrará la sesión teniendo que autenticarse el usuario nuevamente.
Eficiencia
RnF 9. La solución propuesta debe soportar un tiempo de respuesta menor que 5 segundos.
RnF 10. El sistema soportará la conexión simultánea de todos los posibles usuarios, con un promedio de 1000 y un máximo de 3000.
Soporte
RnF 11. El sistema cumplirá con las normas de codificación, conversiones para nomenclatura, bibliotecas de clase definidas para el Sistema de Gestión Universitaria en el documento SGU-NUC-010217_EC.
RnF 11. El sistema contará con toda la documentación definida en el expediente de proyecto asociada a su proceso de desarrollo para las actividades de soporte.
Restricciones de diseño

[16] Java Runtime Environment o JRE es un conjunto de utilidades que permite la ejecución de programas Java.

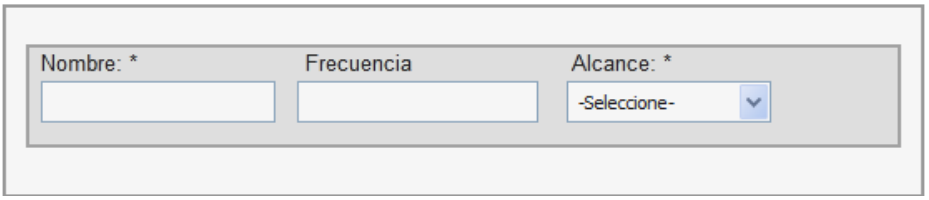
RnF 12. El sistema estará desarrollado con las herramientas definidas para el Sistema de Gestión Universitaria en el documento SGU-INV-010216_8_ATI_v 1.0.
RnF 13. El sistema cumplirá con la arquitectura de información definida para el Sistema de Gestión Universitaria en el documento Vista de Arquitectura de Presentación.
Interfaz
RnF 14. La interfaz de usuario se guiará por la vista de arquitectura de presentación definidas en el documento Pautas de diseño XAUCE-SGU.
Hardware
RnF 15. La comunicación entre el cliente y el servidor de aplicaciones se realiza a través del protocolo HTTPS.
RnF 16 Para la ejecución del sistema se requiere que la PC cliente tenga los siguientes componentes de *hardware*: Pentium 4 o superior, 512 MB RAM y 200 MB disco duro disponible como mínimo.

2.2.4 Descripción de requisitos

Los requisitos son el producto del trabajo final que genera la ingeniería de requisitos. Describe la función y el desempeño de un sistema basado en computadoras y las restricciones que regirán su desarrollo (30). A continuación se muestra en la Tabla 4 la descripción de requisito crear premio. Para ver las descripciones de requisitos en su totalidad, consultar el expediente del proyecto Investigaciones.

Tabla 4: Descripción del requisito Crear premio.

Nº	Nombre	Descripción	Prioridad	Complejidad
RFPR1	Crear premio	El requisito permite crear un nuevo premio. El usuario selecciona del componente Premios del Sistema Gestión de Investigaciones, la opción Premios del menú de funcionalidades Premios. El usuario selecciona la opción Crear en la barra de iconos flotantes. Seguidamente se muestran los datos a llenar, el usuario	Alta	Alta

		entra el nombre del premio, la frecuencia y el alcance del mismo. Luego de llenar todos los campos correctamente selecciona la opción Aceptar y se crea el nuevo premio. Si desea cancelar la acción selecciona la opción Cancelar.		
	Prototipo			
	 		
	Campos	**Tipos de Datos**	**Reglas o Restricciones**	
	• Nombre	• Campo de texto	• Único. • Obligatorio. • Solo admite números y letras. • Admite de 2 a 50 caracteres.	

	• Frecuencia	• Campo de selección	• Selección. • Obligatorio.
	• Alcance	• Campo de texto	• Obligatorio. • Solo admite letras.
	Observaciones	1. El usuario debe estar autenticado en el sistema. 2. Si inserta todos los datos correctamente el sistema muestra el mensaje *¨El elemento ha sido creado satisfactoriamente¨.* 3. En caso que se deje un campo obligatorio vacío se muestra en rojo encima del campo el mensaje *¨Campo requerido¨.* 4. Cuando los campos lleguen a la cantidad máxima de caracteres no se permitirá seguir escribiendo. 5. Si el nombre del premio ya existe, se muestra el mensaje: *"El elemento ya existe¨.* 6. Si se entran menos caracteres de los mínimos establecidos se muestra un mensaje de error: *"Entre al menos 2 caracteres".* 7. Si selecciona la opción Cancelar el sistema muestra el mensaje de confirmación *"¿Está seguro de realizar la acción?".*	

2.3 Descripción de la propuesta de solución

El SGU es un sistema que está compuesto por los sistemas Pregrado, Postgrado, Cooperación, Residencia, Teleformación, Biblioteca, Desarrollo, Tecnologías, Investigaciones, Extensión, Organizaciones y Egreso, que agrupan las diferentes áreas de procesos de la UCI. Por su parte el Sistema de Investigaciones está conformado por los componentes: Eventos investigativos, Estructura investigativa, Publicaciones, Premios y Evidencias como se ilustra en la Figura 1.

El componente de Publicaciones se encarga de validar las publicaciones e imprimir los certificados a los investigadores de la universidad, mientras que el componente Premios se encarga de la gestión de los premios, evaluación y selección de los premiados.

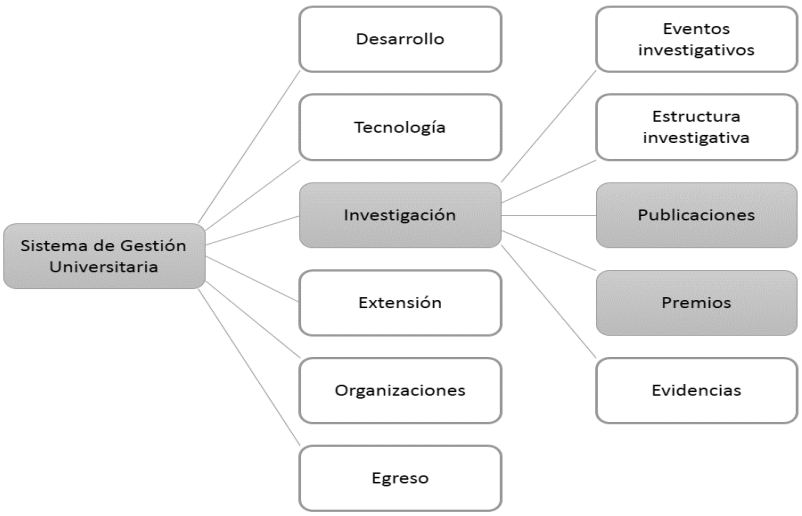


Fig. 1: Mapa de navegación

2.3.1 Publicaciones

El componente Publicaciones estará compuesto por cinco agrupaciones funcionales. La agrupación Fuente admitirá proponer una fuente en caso de que no exista cuando se solicita la certificación de su publicación, la cual será verificada antes de ser añadida al sistema. Además la agrupación Publicaciones permitirá al usuario solicitar la certificación de una publicación de la cual sea autor o coautor, para validar la existencia de dicha publicación el verificador podrá acceder a todos los datos introducidos y tendrá permisos para modificarlos. En todo momento el investigador puede contemplar el estado de su solicitud ya sea mediante el sistema o a través del correo electrónico, pues será notificado cada vez que esta cambie su estado. La agrupación Certificados mostrará un registro de las publicaciones del usuario autenticado y permitirá realizar búsquedas de las publicaciones de otros usuarios. En el caso de sus publicaciones podrá solicitar la impresión del certificado si no ha sido impreso con anterioridad. A través de la agrupación funcional Impresión se listan las solicitudes por imprimir y los certificados ya impresos a la DI para realizar su impresión. Finalmente, la agrupación funcional Configuración permitirá personalizar los elementos de gestión genéricos del componente Publicaciones del SGU. En la Figura 2 se muestra la estructura del componente publicaciones:

Fig. 2: Funcionalidades del componente Publicaciones

La solución garantizará la centralización de los datos, agilizará el proceso de certificación de publicaciones y mantendrá al usuario informado del estado en que se encuentra su solicitud. Para solicitar la certificación de una publicación el investigador debe introducir los datos entre los que se encuentra la fuente, la solución tendrá definido un conjunto de fuentes, pero le permitirá al usuario proponer una nueva fuente en caso de que no exista. Mediante el componente el verificador podrá listar todas las solicitudes para verificar la existencia de la publicación y notificar al usuario del estado de la solicitud. El certificado se imprimirá en caso de ser solicitado por el investigador. Se llevará un registro de las publicaciones donde el investigador podrá consultar sus publicaciones y las de otros usuarios.

2.3.2 Premios

El componente Premios estará compuesto por cinco agrupaciones funcionales. La agrupación Premios, mediante la cual el administrador puede introducir los premios en el sistema asociándole los tipos de premios y las categorías en las que se lanzará la convocatoria. Para cada tipo de premio el administrador definirá el período de entrega de la solicitud y creará las ediciones, especificando la comisión y criterios de evaluación, los que pueden ser asociados al premio en general o para cada categoría en específico. Para cada criterio de evaluación se seleccionará un peso que diferencia la importancia entre cada uno.

La agrupación funcional Solicitudes facilitará a los propuestos al premio la introducción de los datos al sistema para la confección del expediente, los datos se definirán a partir de los requisitos para optar por un

premio, para lo cual el administrador creará el formulario con la información que debe introducir el usuario para solicitar un premio en una categoría específica. Además permitirá verificar a nivel de área y después en la DI los datos introducidos por el usuario permitiendo modificar dichos datos a los responsables del proceso.

Evaluación es otra de las agrupaciones funcionales, a través de la cual se confeccionarán las comisiones, donde para cada una se elegirá un presidente y asociará una importancia referente al peso del criterio de cada miembro. Además mostrará una lista con todas las solicitudes, ya validados sus datos por el responsable de área y la DI, donde a partir de cada solicitud permitirá a cada miembro de la comisión ver el expediente del investigador y emitir una evaluación por cada criterio. Para facilitar la selección de los premiados se generará un listado ordenado con la evaluación total de cada usuario propuesto al premio.

Por último la agrupación Consulta, mostrará un registro de los premios obtenidos por el usuario autenticado y permitirá realizar búsquedas de los premios de otros usuarios, mientras que la agrupación funcional Configuración permitirá personalizar los elementos de gestión genéricos del componente Premios. En la Figura 3 se muestra la estructura del componente Premios:

Fig. 3: Funcionalidades del componente Premios

La solución permitirá a la DI gestionar los premios, las ediciones, las comisiones y las evaluaciones. Cada cierto período se crea una edición de un premio donde serán definidas las categorías, los criterios de evaluación y se asignará la comisión evaluadora. Cuando el usuario realice la solicitud se registrarán los datos para generar el expediente. Los miembros de la comisión podrán registrar las evaluaciones por cada

criterio, dependiendo de esto y de la importancia de cada miembro, el sistema será capaz de proponer los premiados.

Beneficios de la solución:

La utilización de los componentes Publicaciones y Premios permitirá a la Dirección de Investigaciones obtener los siguientes beneficios para la gestión de los premios curriculares y la certificación de publicaciones:

- Mayor organización para el análisis y manejo de la información.
- Información centralizada.
- Acceso a la información para todos los usuarios de la universidad.

La confección del certificado se realiza mediante el sistema, lo que implica que no tenga que pasar por diferentes áreas para su emisión. Además, el sistema cuenta con un registro que permite verificar la existencia de las publicaciones de un investigador, posibilitando imprimir el certificado sólo en caso de ser necesario.

El componente Premios genera un expediente a partir de la solicitud realizada, evitando errores que se generaban anteriormente en su confección. Las comisiones pueden realizar la selección de los premiados mediante el sistema y otorgar individualmente una valoración por cada criterio de evaluación, a partir de las cuales el componente emitirá una propuesta, apoyando de esta forma en la toma de decisiones.

Los aspectos anteriormente expuestos permiten asegurar que a partir del desarrollo de los componente se agilizan los procesos de publicaciones y premios para la gestión de la actividades científicas desarrolladas por la DI en la UCI.

2.4 Descripción de la arquitectura y el diseño

Según Pressman la arquitectura de *software* de un programa o sistema de cómputo es la estructura o las estructuras del sistema, que incluyen los componentes del *software*, las propiedades visibles externamente de los componentes y las relaciones entre ellos (30). Es un conjunto de patrones que proporcionan un marco de referencia necesario para guiar la construcción de un software, permitiendo a los programadores, analistas y todo el conjunto de desarrolladores compartir una misma línea de trabajo y cubrir todos los objetivos y restricciones de los componentes.

2.4.1 Arquitectura

La arquitectura Cliente-Servidor es un modelo de aplicación distribuida, donde las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes. Un cliente realiza peticiones a otro programa, el servidor, que le da respuesta. En esta arquitectura la capacidad de proceso está repartida entre los clientes y los servidores, aunque son más importantes las ventajas de tipo organizativo debido a la centralización de la gestión de la información y la separación de responsabilidades, lo que facilita y clarifica el diseño del sistema.

Cliente: permite al usuario formular los requisitos y pasarlos al servidor. Maneja todas las funciones relacionadas con la manipulación y despliegue de datos, por lo que están desarrollados sobre plataformas que permiten construir interfaces gráficas de usuario, además de acceder a los servicios distribuidos en cualquier parte de una red. Algunas de sus funciones son administrar la interfaz de usuario, interactuar con el usuario, procesar la lógica de la aplicación, hacer validaciones locales, generar requisitos de bases de datos, recibir resultados del servidor y formatear resultados.

Servidor: encargado de atender a múltiples clientes que hacen peticiones de algún recurso administrado por él. Maneja todas las funciones relacionadas con la mayoría de las reglas del negocio y los recursos de datos. Las funciones del servidor son aceptar los requisitos de bases de datos que hacen los clientes, procesar esos requisitos, formatear datos para trasmitirlos a los clientes, procesar la lógica de la aplicación y realizar validaciones a nivel de bases de datos. A continuación se muestra en la Figura 4 la arquitectura Cliente-Servidor:

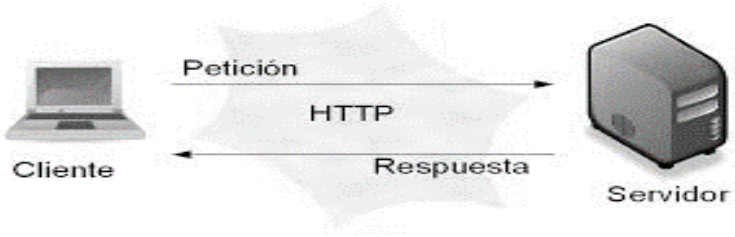


Fig. 4: Arquitectura Cliente-Servidor

2.4.2 Patrón de arquitectura

El patrón Modelo-Vista-Controlador separa los datos de una aplicación, la interfaz de usuario y la lógica de control en tres componentes distintos. Se ve frecuentemente en aplicaciones *web*, donde la vista es la página HTML y el código que provee datos dinámicos a la página, la modelo es el sistema de gestión de base de datos y el controlador representa la lógica de negocio (31).

Modelo: es la representación de la información en el sistema. Trabaja junto a la vista para mostrar la información al usuario y es accedido por el controlador para añadir, eliminar, consultar o actualizar datos.

Vista: presenta un formato adecuado para que el usuario pueda interactuar con el sistema, casi siempre es la interfaz de usuario.

Controlador: es el elemento más abstracto. Recibe, trata y responde los eventos enviados por el usuario o por la propia aplicación. Interactúa como intermediario entre el modelo y la vista.

Funcionamiento del patrón MVC en el marco de trabajo GUUD

En la Figura 5 se muestra el funcionamiento del marco de trabajo que cuenta con un controlador frontal que se encarga de inicializar los recursos necesarios.

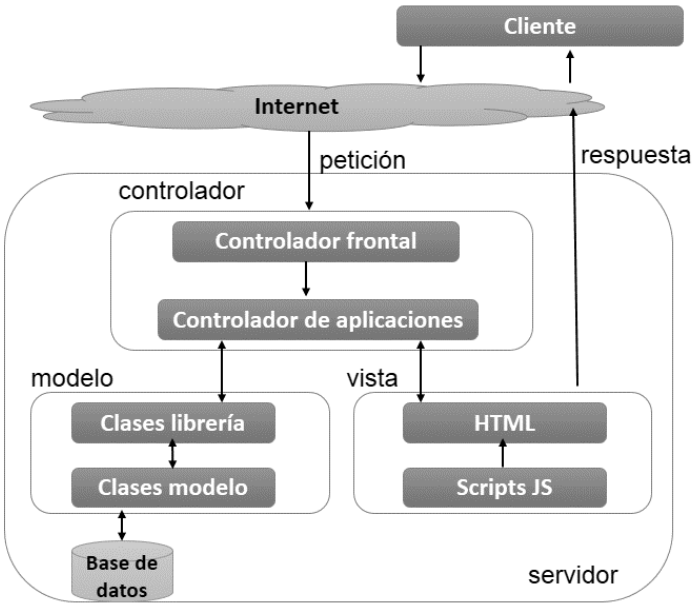


Fig. 5: Funcionamiento del patrón en GUUD

Luego del cliente realizar una petición HTTP, el controlador frontal analiza y determina cuál controlador de aplicación (controlador de una determinada funcionalidad de un componente) debe ser cargado para atender la petición realizada. Cada controlador de aplicación tiene asociada una o varias librerías, y esta a su vez tiene asociada una o varias modelos. Las librerías son responsables de procesar los datos e implementar la lógica del negocio y las clases modelos son las encargadas del acceso a los datos. Cuando es cargado el controlador de aplicaciones, este analiza la petición para determinar si necesita interactuar con la base de datos o solo cargar una vista determinada. En caso que necesite interactuar con la base de datos envía los datos recibidos a las librerías y estas cargan las modelos necesarias para obtener, registrar o actualizar en la base de datos la información solicitada. Luego de ser obtenidos los datos en un proceso inverso al anterior, se retornan al controlador de aplicación. Posteriormente, el controlador carga estos datos a archivos escritos en HTML los cuales pueden incluir llamadas a archivos escritos en *JavaScript* para manejar dinámicamente su contenido. Finalmente, el resultado obtenido de todo este proceso es enviado al navegador *web* como respuesta a la petición inicial.

Ventajas de utilizar este patrón de arquitectura:

- Facilita el mantenimiento.
- Las vistas proveen mayor flexibilidad y agilidad pues se pueden crear múltiples vistas de un modelo, las vistas pueden anidarse y sincronizarse.
- Mayor cohesión ya que cada elemento del patrón está altamente especializado en su tarea, la vista en mostrar datos al usuario, el controlador en las entradas y el modelo en su objetivo del negocio.
- Más claridad de diseño.
- Diseño modular, favoreciendo la reutilización.

2.4.3 Diagrama de despliegue

El diagrama de despliegue ilustra la estructura física del sistema, incluyendo el *hardware*, las relaciones entre el *hardware* y el *software* que se despliega. Puede mostrar servidores, estaciones de trabajo, impresoras, componentes de *software* o *hardware* (32). Este diagrama muestra las relaciones físicas entre los componentes de *hardware* y *software* en el sistema final. Describe la topología del sistema: la estructura

de los elementos de *hardware* y el *software* que ejecuta cada uno de ellos, dicha estructura se muestra en la Figura 6.

Fig. 6: Diagrama de despliegue

Descripción de los elementos e interfaces de comunicación

- **PC cliente:** su función es acceder al sistema e interactuar con él. Al estar la aplicación desarrollada sobre la *web* la máquina cliente necesita disponer de muy pocas prestaciones puesto a que solo necesita un navegador *web* para poder acceder al sistema y realizar las operaciones necesarias.
- **Servidor aplicaciones**: en este nodo se encuentra la capa de presentación del sistema, la cual es accedida por las máquinas clientes a través de un navegador *web*. Contiene además, todas las funcionalidades del sistema.
- **Servidor de base de datos**: es el encargado de almacenar toda la información generada del sistema.
- **Protocolo HTTPS**[17]: protocolo utilizado para la conexión entre las computadoras cliente y el servidor de aplicaciones.
- **Protocolo TCP/IP**: familia de protocolos utilizado para la conexión entre el servidor de aplicaciones y el servidor donde se encuentra ubicada la base de datos.
- **Impresora**: utilizada para imprimir los certificados de publicaciones generados por el sistema.

[17] **HTTPS** *Hypertext Transfer Protocol Secure* (Protocolo seguro de transferencia de hipertexto)

2.4.4 Patrones de diseño

Un patrón de diseño es una descripción de clases y objetos comunicándose entre sí adaptada para resolver un problema de diseño general en un contexto particular (33). Son el esqueleto de las soluciones a problemas comunes en el desarrollo de *software*. En otras palabras, brindan una solución ya probada y documentada a problemas de desarrollo de *software* que están sujetos a contextos similares. Están basados en la recopilación del conocimiento de los expertos en desarrollo de *software*. Clasifican y describen formas de solucionar problemas que ocurren de forma frecuente en el desarrollo.

Patrones generales de *software* para asignar responsabilidades (GRASP)

Los patrones GRASP[18] son parejas de problema/solución con un nombre, que codifican buenos principios y sugerencias relacionadas frecuentemente con la asignación de responsabilidades. Describen la asignación correcta de responsabilidades en el diseño orientado a objetos (34). Se describen los patrones con que cuenta la solución y su aplicación en el desarrollo de los componentes. Para el diseño de la propuesta de solución se tuvo en cuenta los patrones GRASP implementados en GUUD: experto, creador, bajo acoplamiento, alta cohesión y controlador. El marco de trabajo busca un máximo rendimiento y flexibilidad en sus soluciones y pone en práctica estos patrones para lograr un sistema reusable y flexible. A continuación se realiza una descripción de los patrones antes mencionados:

- **Experto:** asignar una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad. Indica que la responsabilidad de la creación de un objeto debe recaer sobre la clase que conoce toda la información necesaria para crearlo. Se evidencia en las clases librerías, que son las que cuentan con la información necesaria para cumplir las responsabilidades sobre los elementos de negocio, ejemplo Edición_lib. Se muestra un ejemplo en la Figura 7:

[18] *General Responsibility Assignment Software Patterns, Patrones Generales de Software para Asignar Responsabilidades.*

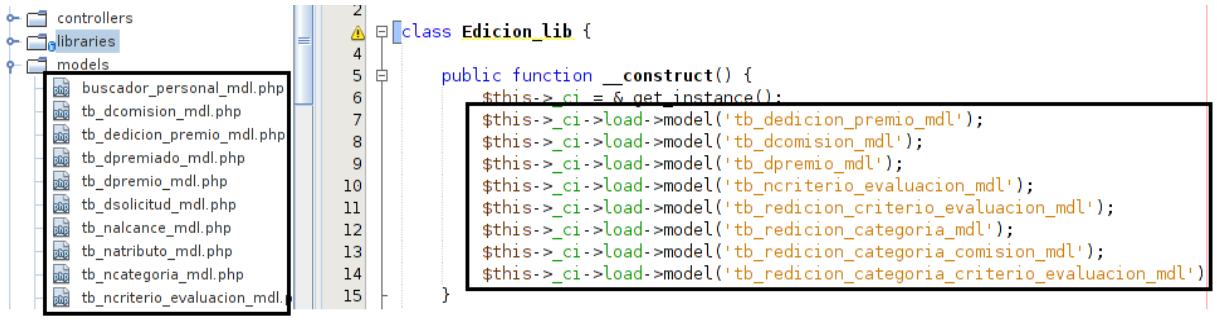


Fig. 7: Patrón experto en la clase Edicion_lib

- **Creador:** asignarle a la clase B la responsabilidad de crear una instancia de clase A, se evidencia en la clase *loader* que es el objeto *load* de las clases controladoras, se encarga de cargar los elementos del marco de trabajo, dígase, librerías y modelos. Se muestra un ejemplo en la Figura 8:

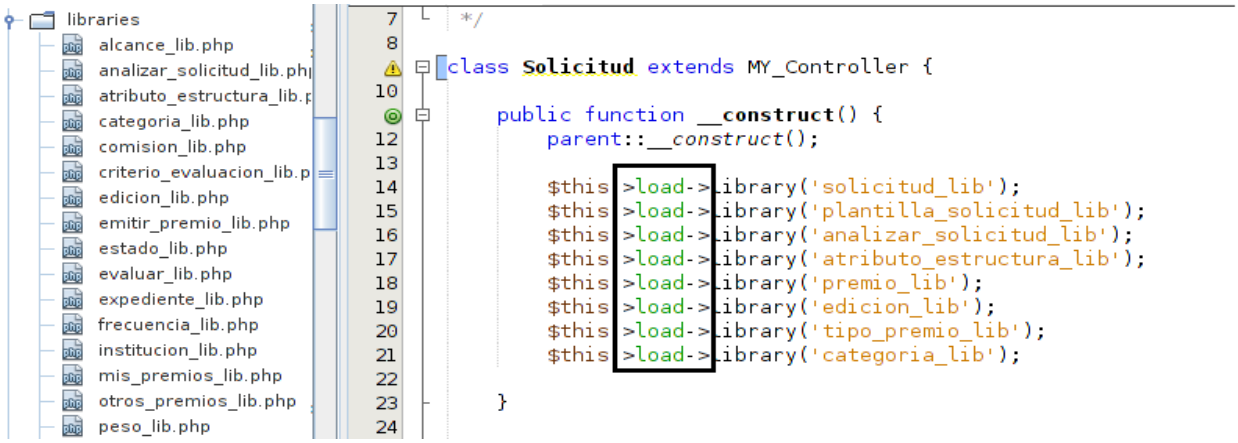


Fig. 8: Patrón creador en la clase Solicitud

- **Controlador:** asignar la responsabilidad del manejo de un mensaje de los eventos de un sistema a una clase. Se evidencia en las clases controladoras que son las encargadas de obtener los datos y enviarlos a las librerías y las vistas. Se muestra un ejemplo en la Figura 9:


```php
<?php
class Premio extends MY_Controller {

    public function __construct() {...13 lines }

    public function index() {...4 lines }

    public function listar() {...9 lines }

    public function modificar() {
        $all_post = $this->input->all_post();
        $token = $this->session->userdata('token');
        $elementos_token = $this->ioc->seguridad->obtenerElementosDadoToken($token);
        if (isset($elementos_token['id_persona'])) {

            $premio = $this->premio_lib->obtenerPremioDadoIdPremio($all_post['id_premio']);
            $tipo_premio = $this->tipo_premio_lib->obtenerTipoPremioDadoIdPremio($all_post['id_premio']);
            $categoria = $this->categoria_lib->obtenerCategoriaDadoIdPremio($all_post['id_premio']);

            $this->template->set_data("premio", $premio);
            $this->template->set_data('arreglo_tipo_premio', $tipo_premio);
            $this->template->set_data('arreglo_categoria', $categoria);

            echo $this->template->render('premio/modificar_premio_view');
        } else {
            throw new Exception_Error('SYS006');
        }
    }
```

Fig. 9: Patrón controlador en la clase Premio

- **Bajo acoplamiento:** asigna una responsabilidad para mantener bajo acoplamiento. El grado de acoplamiento no puede considerarse aisladamente de otros principios como experto y alta cohesión. Es un patrón evaluativo que tiene en cuenta el diseñador al realizar mejoras al diseño.
- **Alta cohesión:** asignar una responsabilidad de modo que la cohesión siga siendo alta. También como el patrón Bajo Acoplamiento es un principio que se deben tener presente en todas las decisiones de diseño.

Los dos patrones anteriormente mencionados se encuentran definidos en la implementación del *framework*. La propia implementación de *CodeIgniter* contiene los dos últimos patrones nivelados, permite el uso de los componentes de forma individual, evidenciando el bajo acoplamiento así como la dependencia entre ellos o alta cohesión.

Patrones GoF

Los patrones GoF (*gang of four*, por sus siglas en inglés), describen las formas comunes en que diferentes tipos de objetos pueden ser organizados para trabajar unos con otros. Tratan la relación entre clases, la combinación de clases y la formación de estructuras de mayor complejidad. Permiten crear grupos de objetos que ayudan a realizar tareas complejas. Estos patrones pueden ser de tres tipos: de creación, estructurales y de comportamiento. Los patrones de creación abstraen la forma en la que se crean los

objetos, permitiendo tratar las clases a crear de forma genérica dejando para más tarde la decisión de qué clases crear o cómo crearlas. Por otra parte los de comportamiento estudian las relaciones entre llamadas entre los diferentes objetos, normalmente ligados con la dimensión temporal. Seguidamente se mencionan estos patrones:

- **Instancia única** (*Singleton*): garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia. Se evidencia en todas las clases controladoras que son instancias únicas, además en la variable IoC para la interacción entre componentes, ejemplo solicitud.php. Se muestra un ejemplo en la Figura 10:

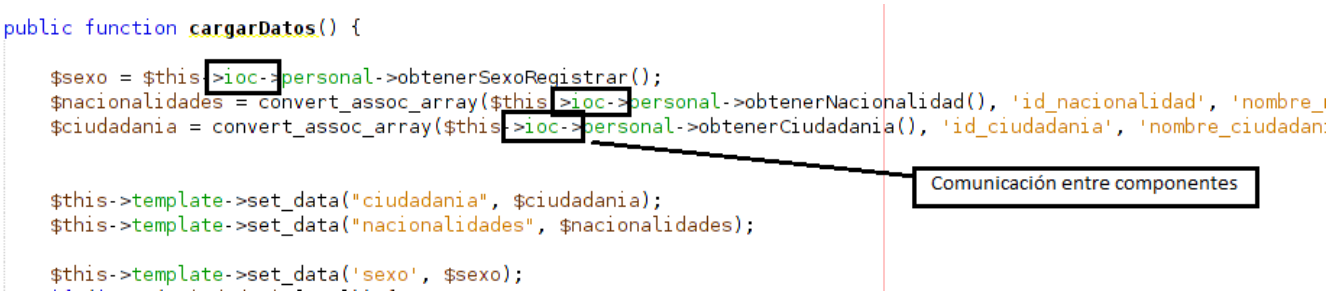


Fig. 10: Patrón instancia única en la clase Solicitud

- **Mediador** (*Mediator*): define un objeto que coordine la comunicación entre objetos de distintas clases, pero que funcionan como un conjunto. Se muestra en las librerías que funcionan como mediadoras entre las clases controladoras y las modelos o acceso a datos, ejemplo solicitar_lib.

- **Observador** (*Observer*): define una dependencia de uno a muchos entre objetos, de forma que cuando un objeto cambie de estado se notifique y actualicen automáticamente todos los objetos que dependen de él, este patrón se evidencia en la clase loader que es el objeto load de las clases controladoras, se encarga de cargar los elementos del marco de trabajo dígase, librerías, modelos y se encarga de actualizar la controladora instanciada.

2.4.5 Patrones de base de datos

El uso de patrones de diseño en una base de datos (BD) permite al usuario crear una BD más fortalecida ya que constituyen una guía que especifica cómo debe ser la misma. El diseño y construcción de una base de datos requiere del mayor esfuerzo y análisis posible, ya que a partir de este diseño es que esta se crea y la calidad con que se obtenga determinará su comportamiento futuro (35).

- **Entidad-Atributo-Valor**

Es la representación de un modelo flexible donde se pueden representar objetos con sus atributos. Es un acercamiento al modelo orientado a objeto representado en el modelo relacional, donde la entidad *Class* representa las clases, la entidad *Attribute* representa los atributos de las clases, por su parte la entidad *Object* representa las instancias de las clases, mientras que la entidad *Value* representa los valores de cada atributo para cada objeto dado (35).

- **Árboles fuertemente codificados**

A cada nivel del árbol se le asocia una entidad. Normalmente constituyen relaciones de uno a mucho. Se utiliza para representar jerarquías donde es bien conocida la estructura y es importante representar la correspondencia, por ejemplo las estructuras organizacionales. Además admite tantos niveles como requiera la jerarquía que se vaya a representar (35), la Figura 11 muestra un ejemplo.

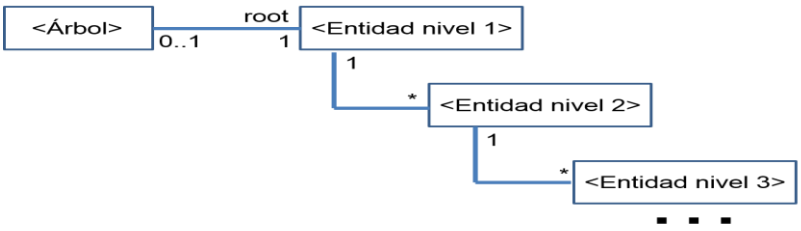


Fig. 11: Árbol fuertemente codificado

- **Llaves subrogadas**

Este patrón es muy utilizado por facilitar la interacción con la BD en un futuro. El mismo plantea que se genere una llave primara única para cada entidad, en vez de usar un atributo identificador en el contexto dado. Normalmente se usan enteros en columnas *identity* o GUID (*Global UniqueIdentifier*) que están demostradas que no se repiten o con una probabilidad extremadamente baja. Esto permite que las tablas sean más fáciles de consultar a partir del identificador, pues todos tienen el mismo tipo en cada una de las tablas (35). Mediante este patrón, fueron generados los identificadores de todas las tablas pertenecientes al modelo de datos que se muestra más adelante.

2.4.6 Modelo de datos

Un modelo de datos es una definición lógica, independiente y abstracta de los objetos, operadores y demás que en un conjunto constituye la máquina abstracta con la que interactúan los usuarios. Los objetos permiten modelar la estructura de los datos. Los operadores permiten modelar su comportamiento (36). El modelo de la base de datos consta de tres fases: diseño conceptual, lógico y físico de la base de datos. La primera fase consiste en la producción de un esquema conceptual que es independiente de las consideraciones físicas. Luego el modelo se refina en un esquema lógico eliminando las construcciones que no se puede representar en el modelo de base de datos escogido. En la tercera fase, el esquema lógico se transforma en un esquema físico para el sistema gestor de base de datos elegido. La fase de diseño físico considera las estructuras de almacenamiento y los métodos de acceso necesarios para proporcionar un acceso eficiente a la base de datos. Los modelos físicos de las bases de datos de los componentes Premios y Publicaciones se encuentran en los Anexos 4 y 5 respectivamente.

2.5 Conclusiones parciales

La descripción de los procesos de gestión de premios y certificación de publicaciones en la UCI permitió su mejor comprensión, así como definir los principales actores en dichos procesos. El uso de técnicas para la obtención de requisitos posibilitó comprender, identificar y describir las exigencias funcionales y no funcionales que deberá cumplir la solución. La utilización de patrones de diseño y arquitectura permitió diseñar una solución robusta, flexible y escalable. A partir del análisis, el diseño realizado y de los artefactos generados quedan sentadas las bases para la implementación y evaluación de la solución propuesta.

Capítulo 3: Implementación y evaluación de la solución

Introducción

En este capítulo se describen los elementos establecidos por el proceso de desarrollo empleado, que responden a la implementación de la solución. Se plantea la solución desarrollada, se definen los estándares de codificación a emplear para el desarrollo de la propuesta y la integración con otros componentes y sistemas. Se realiza el diseño y aplicación de las pruebas que permiten verificar el funcionamiento de la solución propuesta y se analiza los resultados para cada estrategia de prueba.

3.1 Estilos de programación

En la disciplina de implementación se lleva a cabo la producción del software. El equipo de desarrollo se encarga de implementar las funcionalidades, especificadas por el cliente en las historias de usuario, utilizando el lenguaje de programación elegido. Para el desarrollo del sistema se utilizaron dos estilos de programación, la programación orientada a objetos y la programación dirigida por eventos incluidos en el marco de trabajo GUUD.

La programación orientada a objetos (OOP por sus siglas en inglés): es un paradigma de programación que usa objetos y sus interacciones para diseñar aplicaciones y programas de computadora. Está basado en varias técnicas, incluyendo herencia, modularidad, polimorfismo y encapsulamiento (37). El marco de trabajo GUUD al desarrollarse con la tecnología PHP, permite reutilizar el código utilizando el paradigma de programación orientado a objetos que ya posee implementado. En él, cada objeto es responsable de inicializarse y destruirse de forma correcta, no existe la necesidad de llamar explícitamente al procedimiento de creación o de terminación debido a que el marco de trabajo se encarga de esto. Una de las principales características que proporciona es la herencia, lo que permite la reutilización del código y GUUD la utiliza entre los distintos tipos de clases como las modelos y las controladoras.

La programación dirigida por eventos (EDP por sus siglas en inglés): es un paradigma de programación en el que tanto la estructura como la ejecución de los programas van determinados por los sucesos que ocurran en el sistema, definidos por el usuario o que sistema mismos provoque (38). El marco de trabajo GUUD implementa el paradigma de programación dado que la biblioteca *JQuery* lo utiliza. Esta librería es usada para el manejo de las interfaces de usuario y utiliza los principales eventos que provee el navegador

para la interacción de sus componentes. Estos pueden ser desencadenados por el usuario o por otros eventos en el sistema, permitiendo una mayor interoperabilidad usuario-sistema.

3.2 Estándares de codificación

Los estándares de codificación son un conjunto de reglas a seguir por los desarrolladores con el objetivo de establecer un orden y un formato común en el código fuente del *software* en desarrollo. Son de vital importancia durante la etapa de construcción del *software* ya que permite que el personal del proyecto pueda entender de forma fácil el código, garantizándose la organización y estructura del código fuente (39). Con el propósito de estandarizar las nomenclaturas en la implementación de los componentes y obtener un buen producto. Se utilizan los estándares de codificación establecidos por el Departamento de Desarrollo de la DIN.

3.2.1 Indentación, llaves de apertura y cierre, y tamaño de las líneas

Se debe utilizar una indentación sin tabulaciones con un equivalente a cuatro espacios. El uso de las llaves "{}" será en una nueva línea y la longitud de las líneas de código es aproximadamente de 75 y 80 caracteres para mantener la legibilidad del código. A continuación se muestra en la Figura 12 un ejemplo:

```
public function listar()
{
    $premios = convert_assoc_array($this->edicion_lib->obtenerPremios(), 'id_premio', 'nombre_premio', true);
    $this->template->set_data("premio", $premios);
    echo $this->template->render('analizar_solicitud/listar_analizar_solicitud_view');
}
```

Fig. 12: Indentación, llaves de apertura y cierre, y tamaño de las líneas

3.2.2 Convención de nomenclatura

Variables: se rigen por la nomenclatura *camelCase*. Siempre comienzan con minúscula y en caso de nombres compuestos la primera letra de cada palabra comienza con mayúscula. A continuación se muestra un ejemplo en la Figura 13:

```
$autores = array();
$a = array();
$autores = $allPost['asociados_final'];
$presidente = $allPost['id_presidente'];
$idAutor = "";
$marcado = "";
```

Fig. 13: Nomenclatura para variables

Clases: siempre comienzan con mayúscula, en caso de nombre compuesto las palabras se separan con guion bajo "_" y el resto en minúscula. Un ejemplo se muestra en la Figura 14:

```
<?php
class Comision extends MY_Controller {

    public function __construct() {
        parent::__construct();
        $this->load->library("comision_Lib");

    }
```

Fig. 14: Nomenclatura para clases

Funciones: se rigen por la nomenclatura *camelCase*. Siempre comienzan con minúscula y en caso de nombres compuestos la primera letra de cada palabra comienza con mayúscula. Los parámetros son separados por una coma seguida de un espacio entre dos consecutivos. A continuación se muestra un ejemplo en la Figura 15:

```
119     public function formatearFechaDesdePostgres($fecha) {
```

Fig. 15: Nomenclatura para funciones

Ficheros: todo siempre en minúscula y en caso de nombres compuestos se usa el guion bajo, seguido del sufijo definido para cada tipo de fichero.

- **Vistas**: intuitivo y relacionado con el formulario y/o vista que representa. Sufijo *"_view"*. Ejemplo: detalles_estado_view.
- **Modelos**: nombre de la tabla de la base de datos para la cual se crea. Sufijo *"_mdl"*. Ejemplo: tb_nreferencia_mdl.
- **Librerías**: nombre de la clase que representa. Sufijo *"_lib"*. Ejemplo: estado_publicacion_lib.
- **Controladoras**: nombre de la clase que representa. Ejemplo: tipo_publicacion.

3.2.3 Documentación

Los archivos deben de tener su documentación asociada. Cada clase y función se deben describir con el autor, una breve descripción, la categoría en caso de las clases y los parámetros en caso de las funciones. A continuación se muestra un ejemplo en la Figura 16 de documentación de clases.

```
/**
* Clase controladora estado_publicacion
*
* Esta clase funciona como intermediaria entre
* la vista y las librerías
*
* PHP version 5.6.7
*
* @category    Controladora
* @package     Investigaciones
* @author      Marycarmen Díaz Labrador <mlabrador@estudiantes.uci.cu>
* @author      Angel Antonio Sánchez López-Castro <aalopez@estudiantes.uci.cu>
**/
```

Fig. 16: Documentación de clase controladora

3.2.4 Estructura de control

Las estructuras de control utilizan siempre llaves de apertura y cierre “{}”, incluso en situaciones en las que técnicamente son opcionales. Con esto se aumenta la legibilidad del código y se disminuye la probabilidad de errores lógicos. Si las condiciones son muy largas que sobrepasan el tamaño de la línea, se dividen en varias líneas. Entre las estructuras de control y los paréntesis debe de existir un espacio. Incluyen *if*, *for*, *foreach*, *while*, *switch*. En la Figura 17 se muestra un ejemplo de estructura de control.

```
50      public function registrar() {
51          $token = $this->session->userdata('token');
52          $elementos_token = $this->ioc->seguridad->obtenerElementosDadoToken($token);
53          if (isset($elementos_token['id_persona'])) {
54              $this->cargarDatos();
55          } else {
56              throw new Exception_Error('SYS023');
57          }
58          echo $this->template->render('premio/crear_premio_view');
59      }
```

Fig. 17: Estructura de control

3.3 Integración con otros componentes

Los componentes de Publicaciones y Premios forman parte del Sistema de Investigaciones integrado al SGU, estos utilizan varios componentes ya implementados por otros sistemas adyacentes. Todo esto permite la uniformidad en la arquitectura, centralización de archivos e información y la eliminación de redundancias en el código, puesto que cualquier componente en cualquier sistema puede utilizar las funcionalidades implementadas por los demás.

El componente de Publicaciones se integra al componente Eventos, el cual se encuentra desarrollado en el Sistema de Investigaciones.

Eventos: gestiona los eventos científicos realizados en la universidad. Se utiliza para obtener los eventos en caso de que la publicación se genere a través de la participación en una ponencia.

La solución se integra además con el sistema Núcleo del SGU. Con el objetivo de centralizar la información común de los sistemas integrados, dicho núcleo provee funcionalidades horizontales a las demás aplicaciones del SGU. Entre los componentes del Núcleo con los que se integra Publicaciones y Premios se encuentran:

Personal: permite la obtención de toda la información referente al personal de la institución, necesaria para la gestión de las publicaciones y los premios.

Configuración: gestiona la información referente a la división político administrativa de país, permite la obtención de las provincias y municipios del país.

Estructura y composición: gestiona la información referente a toda la estructura administrativa y la jerarquía de la universidad. Permite obtener las áreas administrativas asociadas a los investigadores.

Publicaciones y Premios también se apoyan de los siguientes componentes del Núcleo:

Seguridad: se implementa a nivel central para todo el SGU, garantiza el acceso a la información dados los niveles de privilegio de cada usuario, haciendo uso de la arquitectura sobre la cual está desarrollado el sistema. Además controla no solo el acceso a un componente determinado, sino también a cada sistema que integra el SGU.

Trazas: gestiona todo lo referente a las incidencias de un usuario sobre el SGU, registrando el usuario, la acción realizada y el momento en que se ejecutó.

3.4 Estrategia de pruebas

Una estrategia de prueba de *software* integra los métodos de diseño de caso de pruebas en una serie bien planeada de pasos que desemboca en la eficaz construcción del *software*. Esta debe incluir pruebas de bajo nivel para confirmar la correcta implementación de todos los pequeños segmentos de código fuente, así como pruebas de alto nivel que evalúen las principales funciones del sistema frente a los requerimientos del cliente (30). Para la realización de las pruebas de la solución se definieron estrategias de prueba del *software* a seguir. Se definieron los métodos de pruebas a aplicarse, las técnicas según el nivel y finalmente se evaluaron las consideraciones y criterios de éxito.

3.4.1 Evaluación de requisitos

La metodología de desarrollo de *software* DAC usada en el desarrollo de SGU propone criterios para la evaluación de los requisitos del cliente y del producto, donde se responden varias interrogantes que determinan si el requisito es aprobado o no.

Interrogantes para validar los requerimientos del cliente:

¿El requerimiento es único?

¿El requerimiento es modificable?

¿El requerimiento puede ser probado?

¿El requerimiento no es ambiguo?

¿El requerimiento está completo?

¿El requerimiento puede ser implementado?

¿El requerimiento es congruente con otros requerimientos relacionados?

Técnicas de evaluación de requisitos

Revisiones de requisitos: se realizan reuniones para analizar los requisitos, ayudando a obtener los posibles errores que pueden existir en la especificación de los requisitos del software o para confirmar que los requisitos poseen la calidad deseada.

Generación de casos de prueba: la elaboración de casos de prueba basados en requisitos, permiten analizar el conjunto de entradas, condiciones de ejecución y resultados esperados a partir de los cuales se puede determinar si el requisito de una aplicación es parcial o completamente satisfactorio.

La evaluación de los requisitos permitió identificar deficiencias en las descripciones de requisitos, desechar requisitos que no se consideraba necesarios en la implementación de la solución.

3.4.2 Pruebas de unidad

Las pruebas de unidad se centran en la verificación de los elementos más pequeños del *software* que se puedan probar examinando las estructuras de datos locales. Su objetivo es asegurar que las funcionalidades mantienen su integridad durante los pasos de ejecución de los algoritmos. Para la prueba se utiliza el método de caja blanca o estructural que se basa en un minucioso examen de los detalles procedimentales del código a evaluar. Existen varias técnicas para el diseño de las pruebas de unidad que permiten decidir qué sentencias o caminos se deben examinar con los casos de prueba. La técnica del camino básico se basa en obtener una medida de la complejidad del diseño procedimental de un programa. La medida de complejidad del procedimiento es la complejidad ciclomática y representa un límite inferior para el número de casos de pruebas que se deben realizar para asegurar que se ejecuta cada camino del programa (40).

Para asegurar que se mantiene la integridad de los datos temporales en los pasos de ejecución de un algoritmo, se prueba las estructuras de datos locales. Pretende confirmar que el código funciona de acuerdo con las especificaciones, por lo que se usa el método pruebas de caja blanca que se utiliza si se conoce el funcionamiento interno del producto y la técnica a emplear es camino básico. La misma permite probar las rutas lógicas y la colaboración entre componentes, al proporcionar casos de prueba que ejerciten conjuntos específicos de condiciones, bucles o ambos.

Esta prueba se realiza de forma automática en el sistema, empleando un mecanismo que posee GUUD para su automatización. El marco de trabajo posee la librería o clase especializada *unit_test* para la ejecución de pruebas estructurales. Aunque es sencilla cuenta con una función de evaluación y dos funciones de resultados que permite determinar con certeza si un código específico produce el tipo de dato y resultado esperado. Para ejecutar la prueba se utiliza la línea de código: $this->unit->run(código, resultado esperado, 'nombre de prueba'); donde “código”: es el segmento de código que se desea probar, “resultado esperado”: es lo que debe devolver la evaluación del código que puede ser un tipo de dato o un valor literal

y “nombre de prueba”: es el nombre que se le puede dar a la prueba. La Tabla 5 muestra el diseño de una prueba unitaria, para ver las demás pruebas dirigirse al Anexo 6.

Tabla 5: Diseño de prueba unitaria

Prueba estructural de caja blanca.	**Código de caso de prueba.**
	Autores asociados
Probador:	Angel Antonio Sánchez López-Castro
Tipo de dato esperado:	Array.

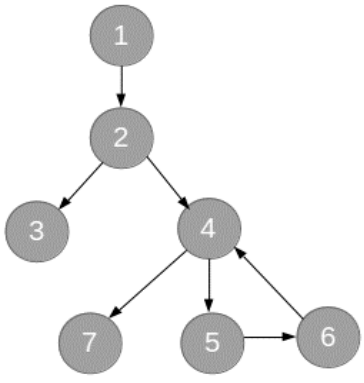
Código al que se aplica:

```
101
    public function obtenerAutoresAsociadosPorPagina($inicio = null, $limite = null,
            $elementoOrdenar = '', $dirOrdenar = 'asc', $params = null) {

105     if (empty($params))
            return array();
107     $datos = $this->_ci->tb_dsolicitud_mdl->obtenerAutoresAsociadosDadoIDSolicitud($params);
108     $persona = array();
109     for ($i = 0; $i < count($datos); $i++) {
110         $obj = new stdClass();
111         $obj->id_persona = $datos[$i]['id_persona'];
112         $obj->nombre_completo = $datos[$i]['nombre_completo'];
113         $obj->numero = $datos[$i]['numero_autor'];
114
115
116         $persona[$i] = $obj;
117     }
118     return $persona;
119 }
```

Complejidad ciclomática:
V(G) = (E − N) + 2 * P
V(G) = (7 − 7) + 2 * 2 = 4
Caminos independientes:
1. 1→2→3
2. 1→2→4→7

Representación del grafo:



Caso de prueba para los caminos básicos

Tipo de dato esperado:	Array.

Función de evaluación:

```
20    public function index() {
21        $this->load->library('Unit_test');
22        echo $this->unit->run($this->certificar_lib->obtenerAutoresAsociadosPorPagina(), array(), 'Obtener autores');
23    }
```

Resultados de la prueba:

Test Name	Obtener autores
Test Datatype	Array
Expected Datatype	Array
Result	Passed
File Name	/var/www/sistema/base/application/investigaciones/certificado_publicaciones/controllers/certificar.php
Line Number	22

Evaluación del caso de prueba:	Satisfactoria

3.4.3 Pruebas de integración

Las pruebas de integración son una técnica sistemática para construir la arquitectura de un *software* a la vez que se aplican las pruebas para encontrar errores asociados a las interfaces (40). Además comprueban que los componentes integrados realmente funcionan juntos, que sean llamados correctamente y que transfieran los datos correctos en el tiempo preciso a través de sus interfaces. Existen dos tipos de integración: incremental y no incremental. La integración no incremental consiste en combinar los componentes como un todo y probar el sistema. La integración incremental se fundamenta en dividir la integración en pequeños incrementos (componentes) y realizar la integración por separado. Propiciando que sea más fácil aislar y corregir los errores.

Se empleó el método incremental ascendente que comienza la construcción y las pruebas en los niveles más bajos de la estructura y combina el siguiente que se debe probar con el conjunto que ya ha sido probado. En la validación de la solución se utiliza el método de caja negra para probar su integración con los sistemas Investigaciones y Núcleo, de este último los componentes Personal y Estructura y composición, apoyándose de Seguridad y Trazas. La Tabla 6 muestra un caso de prueba de integración, los demás están accesibles en el Anexo 7. Las pruebas integración realizadas con los distintos componentes con los que se relacionan Publicaciones y Premios, arrojaron resultados satisfactorios.

Tabla 6: Prueba de integración realizada

Caso de Prueba: Integración de Publicaciones y Premios
Sistema/componente al que se integra: Estructura y composición
Condiciones de ejecución: El componente Estructura y composición debe tener los datos en la base de datos central y debe existir conexión con la misma.
Descripción de la prueba: Comprobar que el componente proporcione la información referente a las estructuras que se van a utilizar para realizar la certificación de publicaciones y la gestión de los premios.
Pasos de ejecución: El controlador realiza una petición a la librería mediante el archivo ioc.xml para obtener las estructuras, gestionados por el componente Estructura y composición.
Resultados esperados: El componente Estructura y composición brinda toda la información solicitada.
Evaluación: Prueba satisfactoria.

3.4.4 Pruebas de aceptación

Las pruebas de aceptación son básicamente pruebas funcionales sobre el sistema completo, ya que tienen como objetivo obtener la aceptación final del cliente antes de la entrega del producto para su utilización. Su ejecución es facultativa del cliente y en el caso de que no se realicen explícitamente, se dan por incluidas dentro de las pruebas del sistema. La ejecución de las pruebas de aceptación requiere un entorno de pruebas que represente el entorno de producción (40). Para el desarrollo de las pruebas se utiliza la técnica alfa donde el cliente hace las pruebas al sistema en un ambiente controlado con el desarrollador como observador del usuario, registrando los errores y los problemas de uso. El desarrollo de la prueba fue satisfactorio, quedando como constancia el acta de aceptación del cliente.

3.4.5 Pruebas de rendimiento

Las pruebas de rendimiento son realizadas luego de que el sistema está completamente integrado, con el objetivo de observar el comportamiento del sistema bajo una cantidad de peticiones esperadas y comprobar que cumpla determinadas tareas en condiciones particulares de trabajo. Se encargan de demostrar que el sistema satisface sus requerimientos y monitorizar los comportamientos en cuanto a tiempo de respuesta de la petición, y otros componentes que se vean afectados por la prueba.

Las pruebas de rendimiento realizadas a la solución se aplicaron con la herramienta Apache JMeter para realizar mediciones exactas y evaluar el rendimiento. Estas fueron realizadas con tres ensayos de 1300,

1400 y 1500 accesos de usuarios respectivamente. Se definió una lista de enlaces a los que se simuló el acceso aleatorio y se recolectaron los datos necesarios para su interpretación. La Tabla 7 muestra un resumen del resultado de las pruebas.

Tabla 7: Resultados obtenidos de las pruebas de rendimiento

Funcionalidad	Usuarios	Tiempo de ejecución				% Error	Rendimiento	
		Mínimo	Máximo	Media	Mediana		Peticiones/seg.	Kb/seg.
Modificar fuente	1300	0	3032	134	1	0.00	1.3	2.14
	1400	0	4141	157	1	0.00	3.6	3.29
	1500	0	4224	210	1	0.00	4.1	3.95
Listar otras fuentes	1300	0	3621	107	1	0.00	2.6	1.74
	1400	0	3504	126	1	0.00	2.9	1.98
	1500	0	4123	197	1	0.00	3.4	3.01
Evaluar propuesta	1300	0	3574	236	1	0.00	2.8	2.46
	1400	0	3954	281	1	0.00	3.7	3.54
	1500	0	4102	312	1	0.00	4.3	4.76
Crear premio	1300	0	2696	119	1	0.00	1.2	1.75
	1400	0	5387	142	1	0.00	3.4	2.53
	1500	0	6291	217	1	0.00	4.2	3.63

- **Usuarios:** indica la cantidad de usuarios haciendo peticiones de manera concurrente.
- **Mínimo:** indica el mínimo de tiempo de ejecución invertido para una petición.
- **Máximo:** indica el máximo de tiempo de ejecución invertido para una petición.
- **Media:** representa el tiempo de ejecución promedio de una petición.
- **Mediana:** significa que el 50% de las peticiones realizadas tardaron menos del valor reflejado.
- **% Error:** indica la relación entre el total de peticiones y el número de peticiones que originaron errores.
- **Rendimiento (Peticiones/seg.):** hace referencia al número de peticiones que el servidor puede procesar en un segundo.

- **Rendimiento (Kb/seg.):** hace referencia a la cantidad de datos que el servidor puede procesar en un segundo.

De manera general las peticiones son procesadas en tiempos relativamente rápido. Para muestras de 1300, 1400 y 1500 no se detectaron errores, lo que significa que el sistema es capaz de soportar mucho más carga de la que se deberá enfrentar en el peor de los casos.

3.4.6 Pruebas de sistema

Las pruebas funcionales tienen por objetivo probar que los sistemas desarrollados cumplen con las funciones específicas para los que han sido creados. Se utiliza el método de caja negra para evaluar funcionalmente la solución donde los probadores se enfocan en el funcionamiento de la interfaz del sistema a partir del estudio de sus entradas y salidas (40). Para confeccionar los casos de prueba de caja negra existen distintas técnicas entre las que se encuentra la técnica de particiones equivalentes. Se basa en identificar las particiones para un sistema o componente. Las particiones de equivalencia son un conjunto de datos segregados por su validez en el sistema, donde cada uno de los miembros de los conjuntos debería ser procesado equivalentemente. Por cada requerimiento funcional del sistema se generó un caso de prueba donde se recogen los datos necesarios para probarlo. Para ver los diseños de casos de pruebas en su totalidad, consultar el expediente del proyecto Investigaciones.

Resultados de las pruebas aplicadas

La pruebas realizadas se llevaron a cabo en tres iteraciones, estas pruebas permitieron detectar un grupo de no conformidades las cuales fueron erradicadas. Fueron detectados en las iteraciones errores en la interfaz, de ortografía y de validación. En la primera iteración fueron detectadas 61 no conformidades para un total de 96 requisitos revisados, en la segunda 23 no conformidades de 136 requisitos y en la última iteración no se detectaron no conformidades. Permitiendo comprobar el correcto funcionamiento de los requisitos funcionales, la calidad de la solución implementada y corregir los errores identificados. A continuación se muestra en la Figura 18 los resultados de las pruebas aplicadas:

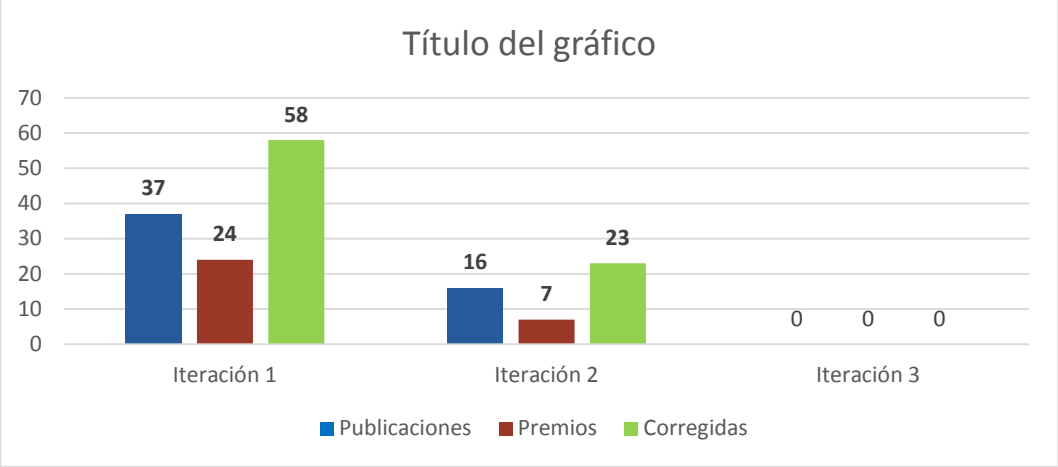


Fig. 18: Resultados de las pruebas funcionales aplicadas a los componentes

3.5 Conclusiones parciales

En este capítulo se abordó la solución desarrollada explicando las funcionalidades que contienen los componentes. La definición de los estándares de codificación y de las técnicas de programación empleadas para la implementación permitió describir el proceso de creación del código de la solución. La integración con otros componentes y sistemas del SGU eliminó posibles redundancias de código facilitando la creación de la solución desarrollada. Las estrategias de pruebas diseñadas posibilitaron demostrar el correcto funcionamiento de los requisitos establecidos, además de que la solución cumple con las especificaciones dadas por el cliente.

Conclusiones Generales

A partir de la investigación realizada, así como del diseño y la implementación de los componentes para la gestión de premios y la certificación de publicaciones en la UCI, se obtuvieron resultados que permiten arribar a las siguientes conclusiones:

- Los sistemas orientados a la gestión de premios y de certificación de publicaciones estudiados no se ajustan a las necesidades del cliente y tecnologías del Sistema de Gestión Universitaria, pero poseen características que sirvieron como base para elaborar la solución.
- La metodología de desarrollo de *software*, permitió guiar el desarrollo de los componentes, documentar la investigación, obtener los artefactos y diseñar la propuesta de solución.
- La implementación de los componentes Publicaciones y Premios para el Sistema de Investigaciones posibilita una mayor organización para el análisis y manejo de los datos, así como la gestión centralizada y accesible de la información en la certificación de publicaciones y entrega de premios curriculares.
- Las pruebas de *software* realizadas a los componentes permitieron corregir las deficiencias detectadas y comprobar el correcto funcionamiento del sistema.

Recomendaciones

Para garantizar el perfeccionamiento progresivo de los componentes de Publicaciones y Premios se recomienda:

- Una vez aprobada la creación de documentos con firma digital, utilizar dicha tecnología para la confección del certificado de publicaciones, con el fin de agilizar el proceso de firma del certificado y evitar su posible falsificación.
- Desarrollar un algoritmo que permita realizar la certificación sin la necesidad de un intermediario, con el objetivo de que el proceso sea automático y por consecuente más rápido.
- Integrar el componente de Premios con el componente de Evidencias del Sistema de Investigaciones, para generar las evidencias necesarias en la creación de los expedientes de las propuestas al premio.

Bibliografía Referenciada.

1. Informáticas, Universidad de las Ciencias. UCI. [En línea] 2012. [Citado el: 11 de marzo de 2015.] http://www.uci.cu/?q=mision.

2. —. Universidad de las Ciencias Informáticas. [En línea] 2012. [Citado el: 21 de marzo de 2015.] http://www.uci.cu/?q=investigaciones.

3. Zayas, Carlos Alvarez de. *Metodología de la Investigación Científica.* Santiago de Cuba : Academia, 1995. pág. 26.

4. —. *Metodología de la Investigación Científica.* Santiago de Cuba : Academia, 1995. pág. 24.

5. Meza, Mario. Organización de Estados Iberoamericanos. [En línea] 2006. [Citado el: 12 de Noviembre de 2014.] http://www.oei.org.co/fpciencia/art07.htm#Intro.

6. GARGANTINI, MARIO y BERSANELLI, MARCO. *Sólo el asombro conoce.* s.l. : Ediciones Encuentro, S.A., 2006.

7. RAE. Real Academia Española. [En línea] 2014. [Citado el: 13 de 11 de 2014.] http://www.rae.es/.

8. Yuste, Adrian. Asociación Española de Normalización y Certificación. *AENOR.* [En línea] 2010. [Citado el: 12 de 11 de 2014.] http://www.aenor.es/aenor/certificacion/procesos/proceso_certificacion_aenor.asp.

9. Universidad Católica Boliviana San Pablo. *Facultad de Ciencias Exactas e Ingeniería.* [En línea] 2012. [Citado el: 13 de 11 de 2014.] http://cisco.ucb.edu.bo/index.php?id=116.

10. Definicion.de. [En línea] 2008. [Citado el: 19 de 11 de 2014.] http://definicion.de/metodologia/.

11. Méndez, Alelí Sánchez. *Proceso de Desarrollo de Software, Metodología DAC.* 2013.

12. JACOBSON, Ivar, RUMBAUGH, James y BOOCH, Grady. *El preoceso unificado de desarrollo de software.* 2000. 84-7829-036-2.

13. Alvarez, Miguel Angel. desarrolloweb.com. [En línea] 23 de 11 de 2009. [Citado el: 19 de 11 de 2014.] http://www.desarrolloweb.com/articulos/codeigniter.html.

14. EcuRed. ECURED. [En línea] 2014. [Citado el: 19 de 11 de 2014.] http://www.ecured.cu/index.php/JQuery.

15. Oracle Corporation, affiliates and/or its. Bienvenido a NetBeans y www.netbeans.org. [En línea] 2013. [Citado el: 19 de 11 de 2014.] https://netbeans.org/index_es.html.

16. Guía-Ubuntu. GUÍA DOCUMENTADA PARA UBUNTU. [En línea] 10 de 03 de 2008. [Citado el: 20 de 11 de 2014.] http://www.guia-ubuntu.com/index.php/PgAdmin_III.

17. Martínez, Rafael. PostgresSQL-es. [En línea] 2 de 10 de 2010. [Citado el: 19 de 11 de 2014.] http://www.postgresql.org.es/sobre_postgresql.

18. Gonzalez, Guillermo. Ciberaula. [En línea] 2013. [Citado el: 20 de 11 de 2014.] http://linux.ciberaula.com/articulo/linux_apache_intro/.

19. Font, Meri. MeriFont Formación 2.0. [En línea] 02 de 04 de 2012. [Citado el: 20 de 11 de 2014.] http://www.merifont.com/pencil-project-crea-tus-prototipos-web/2012/04/02/.

20. Orallo, Enrique Hernández. El Lenguaje Unificado de Modelado(UML). [En línea] [Citado el: 19 de 11 de 2014.] http://www.disca.upv.es/enheror/pdf/ActaUML.pdf.

21. Figueroa, Viridiana. Scribd. [En línea] [Citado el: 19 de 11 de 2014.] http://es.scribd.com/doc/50288837/Caracteristicas-de-PHP.

22. Valdés, Damian Pérez. Misterios del WEB. [En línea] 03 de 07 de 2007. [Citado el: 25 de 11 de 2014.] http://www.maestrosdelweb.com/que-es-javascript/.

23. Vaquero, Miguel. deciencias.net. *Web Docente Departamental.* [En línea] 2010. [Citado el: 25 de 11 de 2014.] http://www.deciencias.net/disenoweb/elaborardw/paginas/intro_html.htm.

24. Marco, Bartolomé Sintes. MCLIBRE-Material Corricular Libre. [En línea] 26 de 05 de 2014. [Citado el: 25 de 11 de 2014.] http://www.mclibre.org/consultar/xml/lecciones/xml_quees.html.

25. Escofet, Carmen Martín. UOC. *Universidad de Catalunya.* [En línea] [Citado el: 26 de 11 de 2014.] http://ocw.uoc.edu/computer-science-technology-and-multimedia/bases-de-datos/bases-de-datos/P06_M2109_02149.pdf. P06/M2109/02149.

26. Wium Lie, Håkon y Bos, Bert. *Cascading Style Sheets, designing for the Web.* Segunda Edición. New York : Addison Wesley, 2010. ISBN 0-201-59625-3.

27. Davenport, Tomas Hill. *Process innovation: reengineering work through information technology.* Boston : s.n., 2010. ISBN: 0875843662.

28. Sommerville, Ian. *Ingeniería de software.* 6ta. Edición. Prentice-Hall : s.n., 2002. ISBN 970-26-0206-8.

29. —. *Ingeniería de Software.* [ed.] Miguel Martín Romo. 7ma. Edición. Madrid : PEARSON EDUCASION, S.A., 2005. ISBN: 84-7829-074-5.

30. Pressman, Roger. *Ingeniería de Software: Un Enfoque Práctico.* 6ta. Edición. s.l. : McGraw-Hill, 2005. pág. 900. ISBN: 9701054733.

31. Mouzo Lema, Juan y Makedonsky, Mariano. *Flash: desarrollo profesional.* Buenos Aires : Fox Andina S.A, 2011. pág. 320. ISBN: 978-987-1857-00-5.

32. Kendall, Kenneth E y Kendall, Julie E. *Análisis y diseño de sistemas.* [ed.] Guillermo Trujano Mendoza, Miguel Gutiérrez Hermandez y Enrique Tejedo Hernandéz. sexta edición. s.l. : Person Educación de México S.A, 2005. pág. 752. ISBN: 970-26-0577-6.

33. Gamma, Erich, y otros. *Desing Patterns: Elements of Reusable Object-Oriented Software.* s.l. : Addison Wesley, 1995. ISBN:0-201-63361-2.

34. Larman, Craig. *UML y Patrones Introducción al análisis y diseño orientado a objetos.* [trad.] Luz Maria Henhndez Rodriguez y Humberto Cárdenas Anaya. México : PRENTICE HALL, 1999. pág. 536. ISBN: 970-17-0261-1.

35. datos, Patrones de diseño de base de. EVA. [En línea] 2013. [Citado el: 23 de Febrero de 2015.] http://eva.uci.cu/file.php/180/2._Clases/Tema_1/Materiales_basicos/4.Patrones_de_diseno_de_BD.pdf.

36. Pons, Olga, y otros. *Introducción a los sistemas de base de datos.* [ed.] Clara de la Fuente Rojo. septima edición. Madrid : Prentice Hall, 2005. ISBN: 978-84-9732-396-3.

37. Álvares, Miguel Ángel. Desarrollo web.com. [En línea] 2011. [Citado el: 30 de Enero de 2014.] http://www.desarrolloweb.com/articulos/codeigniter.html.

38. García, Victor. Programación en Java. *Programación en Java.* [En línea] 13 de diciembre de 2011. [Citado el: 24 de abril de 2014.] http://programarjava.wordpress.com/2011/12/13/programacion-orientada-a-eventos/.

39. MICROSOFT. Revisiones de código y estándares de codificación. [En línea] 2014. [Citado el: 21 de Abril de 2014.] http://msdn.microsoft.com/es-es/library/aa291591%28v=vs.71%29.aspx.

40. Pressman, Roger. *Ingeniería de Software, un enfoque práctico.* Séptima Edición. New York : McGraw-Hill Companies, 2007. ISBN 978-0-07-337597.

41. Zayas, Carlos Alvarez. *Metodología de la Investigación Científica.* Santiago de Cuba : Academia, 1995. pág. 38.

42. MARIO GARGANTINI, MARCO BERSANELLI. *Sólo el asombro conoce.* s.l. : Ediciones Encuentro, S.A., 2006.

43. Gallardo, Marcelo Andrés Saravia. *Metodología de Investigación Científica.* 2006.

44. IMNC. Instituto Mexicano de Normalización y Certificación, A. C. . [En línea] 2013. [Citado el: 13 de 11 de 2014.] http://www.imnc.org.mx/quienes_somos.html.

45. RevistaCiencias.com, Publicaciones Científicas. Publicaciones Científicas RevistaCiencias.com. [En línea] 2009. http://www.revistaciencias.com/.

46. Ambiente, Agencia de Medio. mediambiente.cu. [En línea] 2012. [Citado el: 14 de 11 de 2014.] http://www.medioambiente.cu/organigrama.asp#top.

47. Ambiente, Ministerio de Ciencia Tecnología y Medio. Certificación de Publicaciones Seriadas Científico-Tecnológicas. [En línea] 2013. [Citado el: 14 de 11 de 2014.]

http://blogs.sld.cu/reumatologia/files/2014/08/CERTIFICACI%C3%93N-DE-PUBLICACIONES-SERIADAS-CIENT%C3%8DFICO-TECNOL%C3%93GICAS.pdf.

48. UCVL. Universidad Central ¨Marta Abreu¨ de las Villas. [En línea] 2013. [Citado el: 14 de 11 de 2014.] http://www.uclv.edu.cu/es.

49. EcuRed. ECURED. [En línea] 2014. [Citado el: 19 de 11 de 2014.] http://www.ecured.cu/index.php/Metodologias_de_desarrollo_de_Software#Extreme_Programming_.28XP.29.

50. —. EcuRed. [En línea] [Citado el: 20 de 11 de 2014.] http://www.ecured.cu/index.php/Ubuntu#Caracter.C3.ADsticas.

51. JACOBSON, Ivar, RUMBAUGH, James y BOOCH, Grady. *El proceso unificado de desarrollo de software.* 2000. 84-7829-036-2.

52. FirmaProfesional. [En línea] Enero de 2002. [Citado el: 19 de Febrero de 2015.] https://www.firmaprofesional.com/esp/.

53. *Sistema informático para la certificación de publicaciones.* Linares Armas, Dayamy, Serralvo Cala, Miriam y Valdés Parada, Luisa Elena. Universidad y Sociedad : s.n., Octubre de 2014. ISSN: 2218-3620.

Bibliografía Consultada

1- SOMERVILLE, Ian. Ingeniería del Software. [En línea]. 7a. ed. Madrid: Pearson Education S.A, 2005. 661p. [En línea], [ref. de 3 de febrero de 2015]. Disponible en web: <http://books.google.com.cu/books?id=gQWd49zSut4C&pg=PA80&dq=Herramientas+Case&hl=es&sa=X&ei=uOG3T-_cE8jpgged2PSiCg&ved=0CEQQ6AEwAw#v=onepage&q=Herramientas%20Case&f=false ISBN: 84-7829-074-5>.

2- GUTIÉRREZ, R. H. G. Y. J. M. G. Conocimiento, innovación y desarrollo. San José, Costa Rica 2011, vol. 1ra edición, nº p. 290. Disponible en: http://catedrainnovacion.ucr.ac.cr/librocid.pdf.

3- HERNÁNDEZ, H. La importancia de un Sistema de Información en las empresas. 5 de enero de 2015 2012, nº Disponible en: http://hectorhernandezadm.blogspot.com/.

4- DEFINICIÓN.DE. Definición de reporte. 26 de enero de 2015 2008, nº Disponible en: http://definicion.de/solicitud.

5- VIDAL, Y. G. Documento de Arquitectura de Software del proyecto. 5 de febrero de 2015 nº.

6- INC, C. E. L. CODEIGNITER. 30 de enero de 2013 2013, nº Disponible en: http://codeigniter.com/.

7- PROJECT, J. T. J. The jQuery Project. 3 de febrero de 2015 2010, nº Disponible en: http://docs.jquery.com/Release:jQuery_1.3.2.

8- GROUP, P. G. D. A brief history of PostgreSQL. 2 de febrero de 2015 2013, nº Disponible en: http://www.postgresql.org/docs/9.4/interactive/intro-whatis.html.

9- CIBERAULA. Una introducción a Apache. 7 de febrero de 2015 2010, nº Disponible en: http://linux.ciberaula.com/articulo/linux_apache_intro/.

10- Espinosa Moscoso, Ibrael. Módulo de premios y publicaciones del sistema de Ciencia Tecnología e Innovación en la Universidad de las Ciencias Informáticas. 12 de abril de 2015 2013, nº Disponible en: http://repositorio_institucional.uci.cu/jspui/bitstream/ident/TD_03206_10/1/TD_03206_10.pdf.

11- PARADIGM, V. UML tool, business process modeler and database designer for software development team. 3 de febrero de 2015 nº Disponible en: http://www.visual-paradigm.com.

12- PROJECT, P. Evolus Pencil. 3 de febrero de 2013 2012, nº Disponible en: http://pencil.evolus.vn/.

13- MAESTROSDELWEB. ¿Qué es JavaScript? 4 de febrero de 2015 nº Disponible en: http://www.maestrosdelweb.com/editorial/%C2%BFque-es-javascript/.

14- LIBROSWEB.ES. Introducci'on a CSS. 3 de marzo de 2013 2013, nº Disponible en: http://www.librosweb.es/css/.

15- ECURED. HTML. 5 de marzo de 2015 nº Disponible en: http://www.ecured.cu/index.php/HTML.

16- WALSH, N. A Technical Introduction to XML. 3 de febrero de 2015 1998, nº Disponible en: http://www.xml.com/pub/a/98/10/guide0.html.

17- ÁLVAREZ, R. Qué es y para qué sirve SQL. 4 de marzo de 2015 2001, nº Disponible en: http://www.desarrolloweb.com/articulos/262.php.

18- PRESSMAN, R. S. Ingeniería del software, un enfoque práctico 5ta edición. nº

19- ROMERO, G. M. P. Metodología ágil para proyectos de software libre. Junio 2008, Ciudad de La Habana 2008, nº

20- SCHMULLER, J. UML en 24 Horas. 3 de febrero de 2015 nº Disponible en: http://www.scribd.com/doc/38392190/UML-en-24-Horas.

21- EVA. Patrones de diseño de base de datos. 23 de abril de 2015 2013, nº Disponible en: http://eva.uci.cu/file.php/180/2._Clases/Tema_1/Materiales_basicos/4.Patrones_de_diseno_de_BD.pdf